Big Python

Andreas Schreiber <Andreas.Schreiber@dlr.de>

Python Unconference Hamburg, 29.11.2014

DLR

Vorstellung

Wissenschaftler,
Abteilungsleiter

DLR Deutsches Zentrum für Luft- und Raumfahrt
German Aerospace Center

Co-Gründer,
Geschäftsführer

medando

Communities

Quantified Self Meetup Cologne

py cologne

PyCon. DE 2013

PyHPC2014

Big Python
Themen

Big Number of Devices

- Internet of Things
- Smartphones

Big Computers

- High Performance Computing

Big Applications

- „Killer“-Applikationen in Wissenschaft und Technologie

Big Number of Devices

- Internet of Things
- Smartphones

DLR

Internet der Dinge
Internet of Things

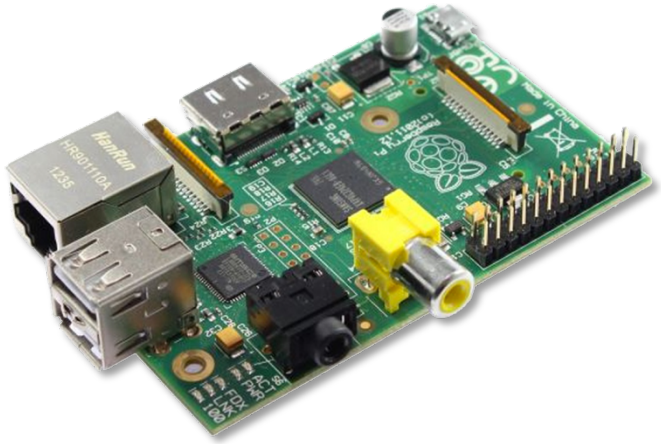
Milliarden an Geräten, Sensoren und Chips

- Verbundene physikalische Objekte (oder deren virtuelle Repräsentation)
- Verbunden über das Internet
- Eindeutig identifiziert
- Sie interagieren miteinander

Geräte im Internet der Dinge

Die „Dinge“ sind

- Embedded Systeme
- Sensoren
- Aktuatoren



In unseren Lebenswelten

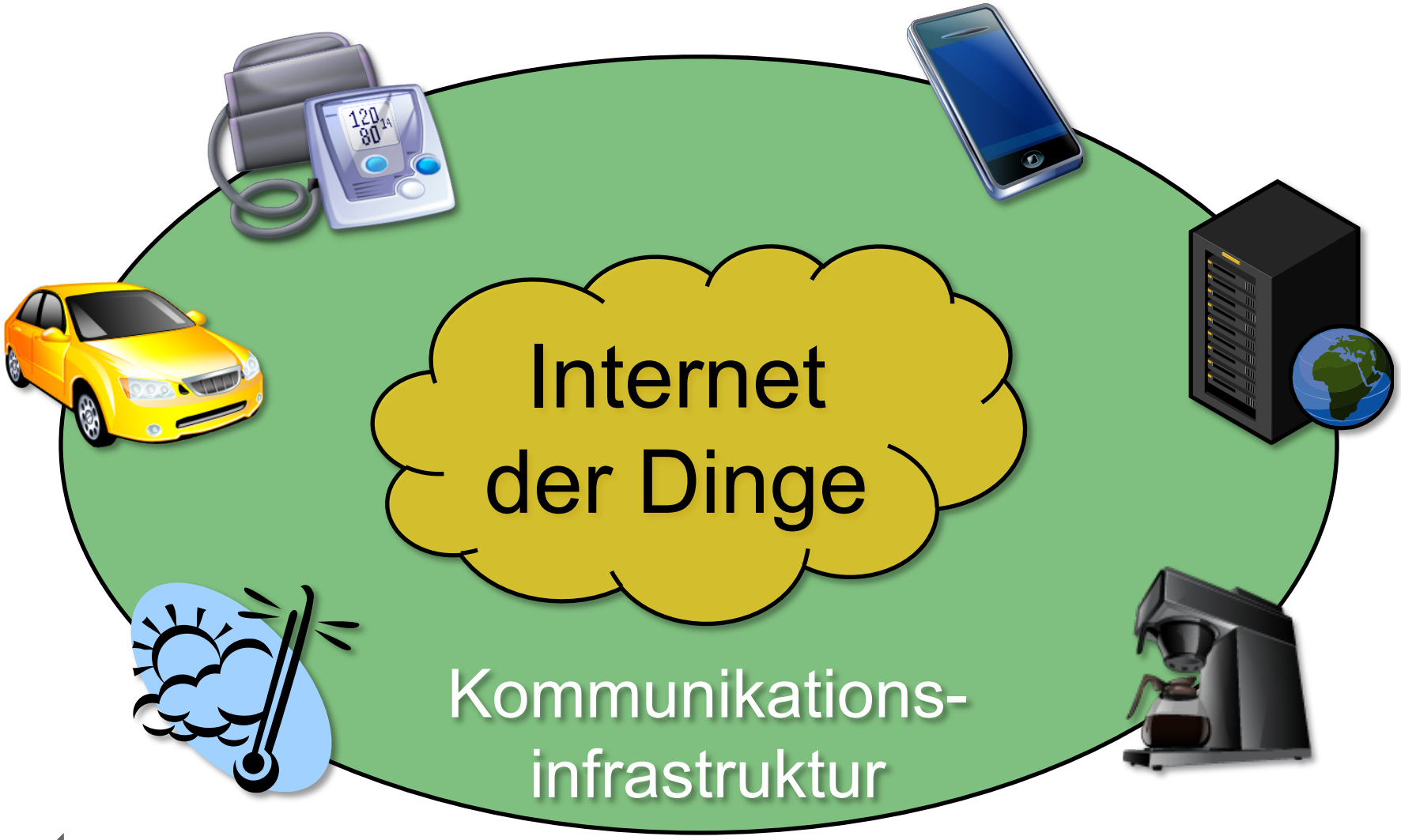
- Smart Home
- Connected Car
- Wearables
- …

Wie groß ist „Big“?
Wachstum

Die Anzahl der mit Internet verbundenen Geräte steigt täglich

50.000.000.000 „Dinge“ bis 2020

Kommunikation

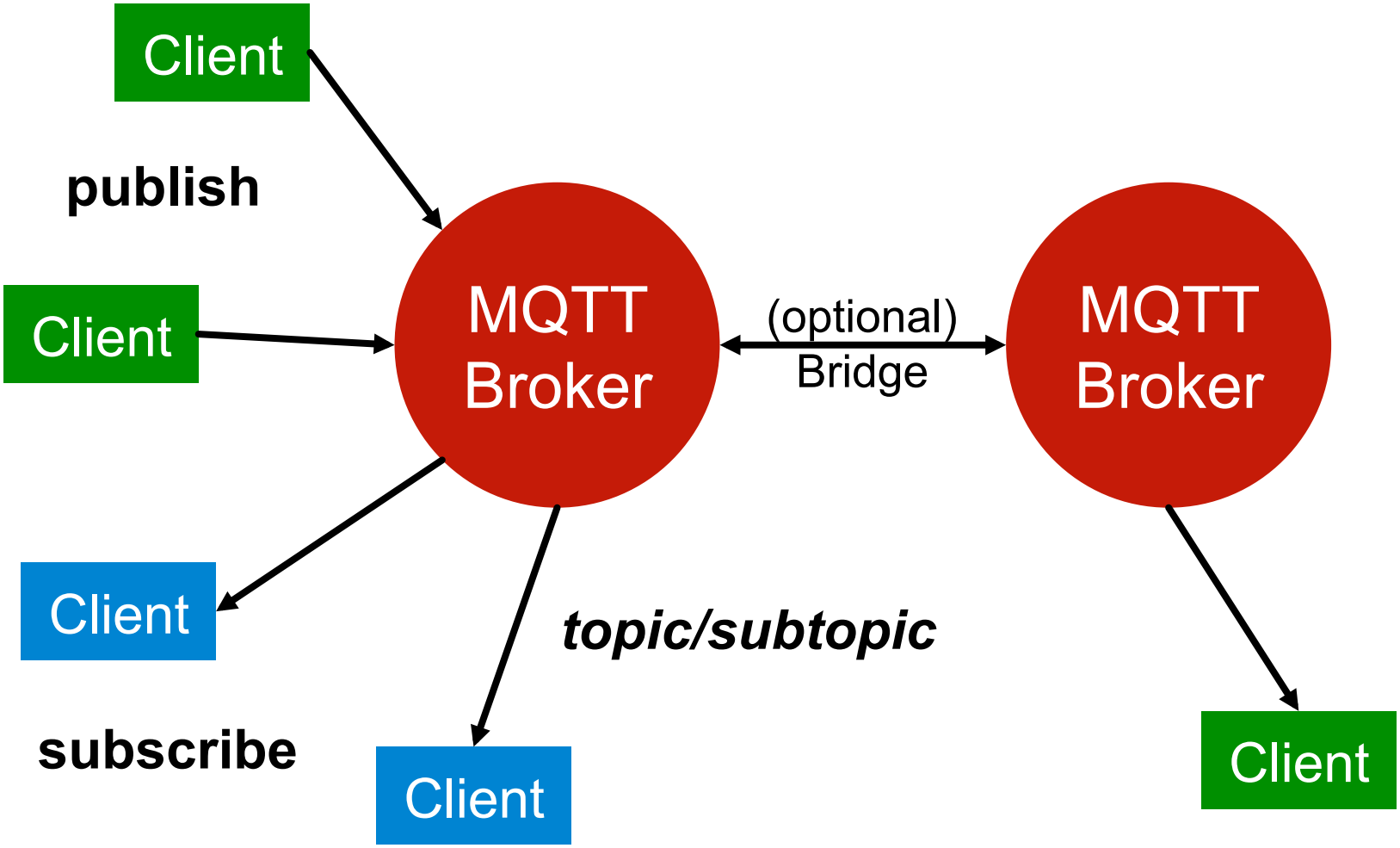


Ein Kommunikationsprotokoll
MQTT

MQ Telemetry Transport

- Machine-to-machine (M2M) connectivity protocol
- Publish/Subscribe-Messaging
- Rechnet mit unzuverlässigen Netzwerken mit geringer Bandbreite und hoher Latenzzeit
- Rechnet mit Clients mit geringer Rechenleistung
- Erlaubt hohen Quality-of-Service, falls das Netzwerk es erlaubt
- Einfach zu implementieren

MQTT
Broker


Client
publish
Client
MQTT Broker
(optional) Bridge
MQTT Broker
Client
topic/subtopic
subscribe
Client
Client

MQTT
Topics

Messages in MQTT werden auf „Topics“ veröffentlicht

- Keine Konfiguration notwendig, einfach auf dem Topic veröffentlichen
- Topics sind hierarchisch mit „/“ als Trenner

```
my/home/temperature/kitchen

my/home/temperature/livingroom

my/server/temperature
```

MQTT
Implementierungen

Server/Broker

- IBM Websphere MQ
- RSMB
- Eclipse Paho
- MQTT.js
- Apache ActiveMQ
- RabittMQ
- HiveMQ

Bibliotheken für

- C/C++
- Java
- Python
- Perl
- PHP
- Ruby
- …

`http://mqtt.org/wiki/software`

MQTT mit Python
Eclipse Paho

Python Client-Modul

- Eine einzelne Datei, reine Python-Implementierung
- Veröffentlichen und Empfangen von Messages
- Callbacks
  - Connect
  - Disconnect
  - Publish
  - Message
  - Subscribe

https://eclipse.org/paho

MQTT mit Python
Subscribe

```
import paho.mqtt.client as mqtt

def on_message(mosq, obj, msg):
    print(msg.topic + ' ' + str(msg.payload))

mqtt_client = mqtt.Client()
mqtt_client.on_message = on_message

mqtt_client.connect('test.mosquitto.org')
mqtt_client.subscribe('#', 0) # all topics

return_code = 0
while return_code == 0:
    return_code = mqtt_client.loop()
```

MQTT mit Python
Publish

```
import paho.mqtt.client as mqtt

mqtt_client = mqtt.Client()

mqtt_client.connect('test.mosquitto.org')

mqtt_client.publish('python/demo',
                    'hello world', 1)
```

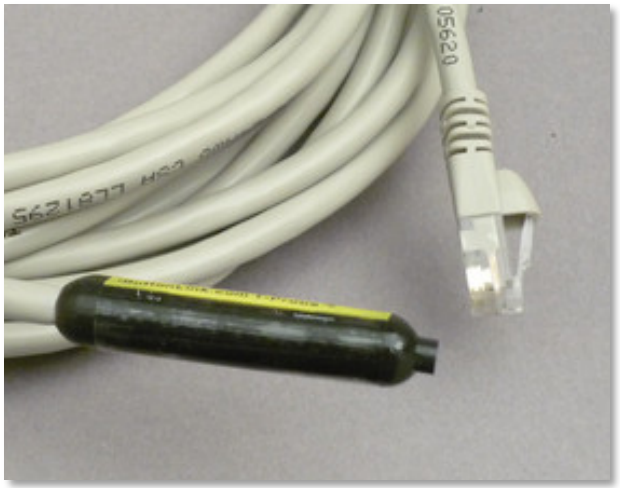
MQTT Anwendungsbeispiel
Heimautomatisierung mit Raspberry Pi

Messdaten mit Sensoren via 1-Wire

- 1-Wire: Einkabel-Bussystem, niedrige Geschwindigkeit
- Sensoren für Temperatur, Spannung, Licht, Feuchtigkeit, …

Eclipse Paho auf Raspberry Pi installieren

- `apt-get install mosquitto`



Messwerte von 1-Wire-Sensoren

- Mehrere Lösungen für Python

Temperatur veröffentlichen
OWFS: One Wire File System

```
import time
import os
import paho.mqtt.client as mqtt

file_name = os.path.join('/', 'mnt', '1wire',
                         '10.67C6697351FF', 'temperature')

mqtt_client = mqtt.Client('home-temperature')
mqtt_client.connect('test.mosquitto.org')

while 1:
    file_object = open(file_name, 'r')
    temperature = '%sC' % file_object.read()
    mqtt_client.publish('home/demo/temperature', temperature, 1)
    mqtt_client.loop()
    time.sleep(5)
    file_object.close()
```

Hardware für das Internet der Dinge
WunderBar

Relayr WunderBar

- IoT Starter Kit
- Verschiedene Sensoren
- WiFi und Bluetooth LE
- SDKs und APIs
  - Android
  - iOS/OSX
  - Python
  - Web/Javascript

https://relayr.io

Bild: relayr.io

WunderBar
Hardware

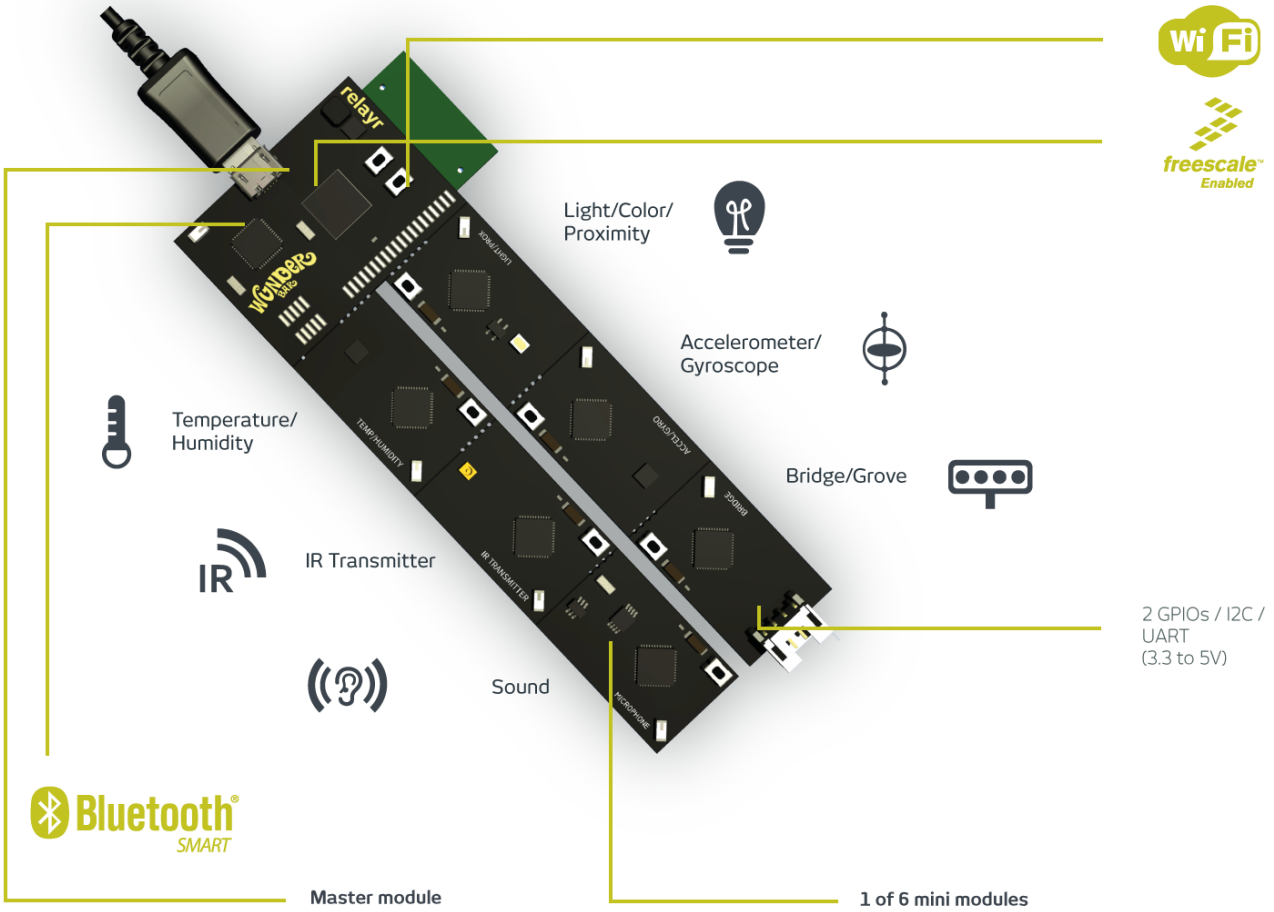
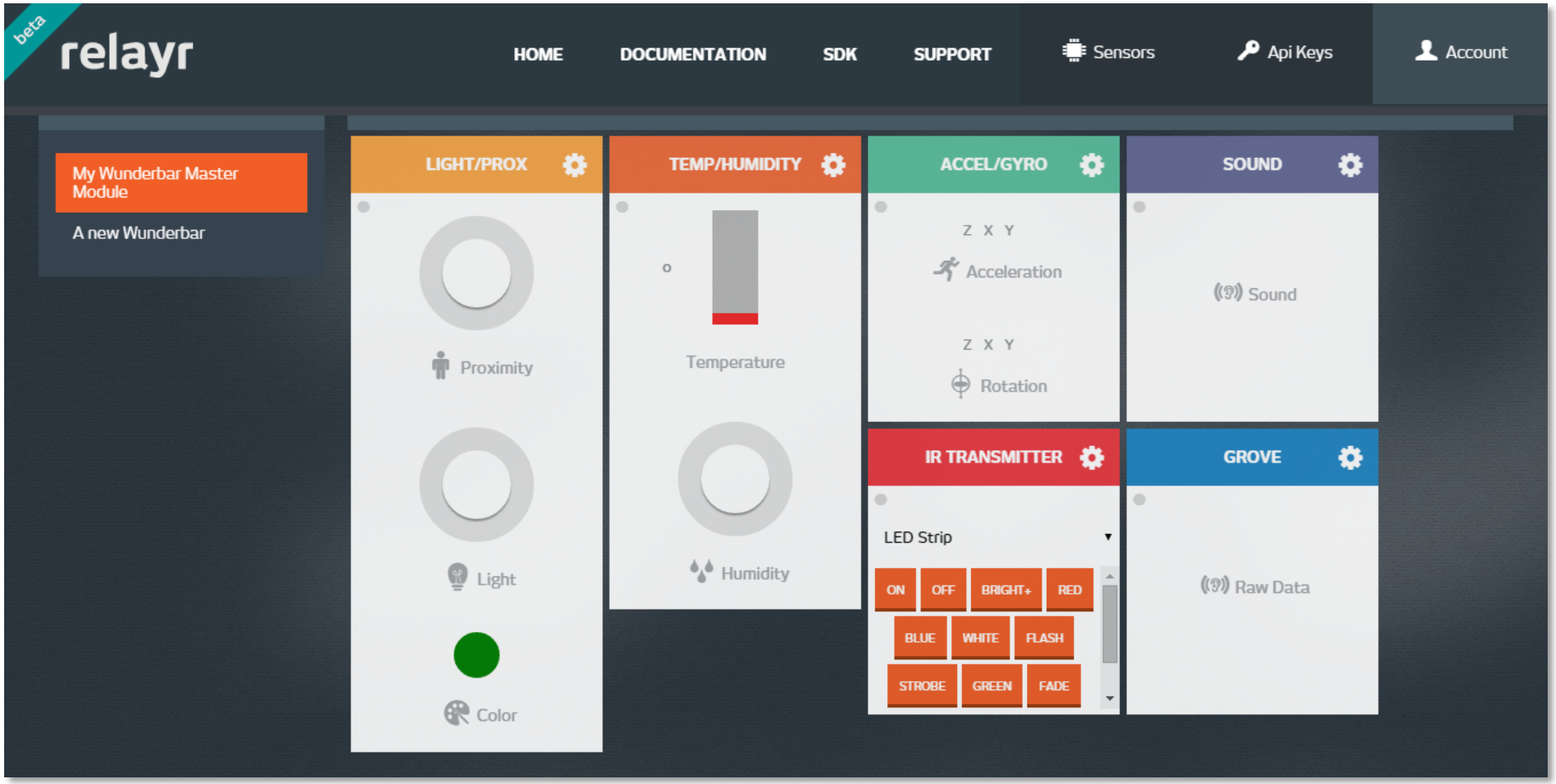


Bild: relayr.io

WunderBar
Cloud Service und Web-Oberfläche



WunderBar
Python SDK

```
from relayr import Client

client = Client(token='XXX')
device = client.get_device(id='XXX')

device.switch_led_on(True)
```

WunderBar
Python SDK

```
import time
from relayr import Client

def callback(message, channel):
        print(repr(message), type(message))

client = Client(token='<XXX>')
device = client.get_device(id='<XXX>').get_info()

user = client.get_user()
conn = user.connect_device(device, callback)
conn.start()
time.sleep(10)
conn.stop()
```

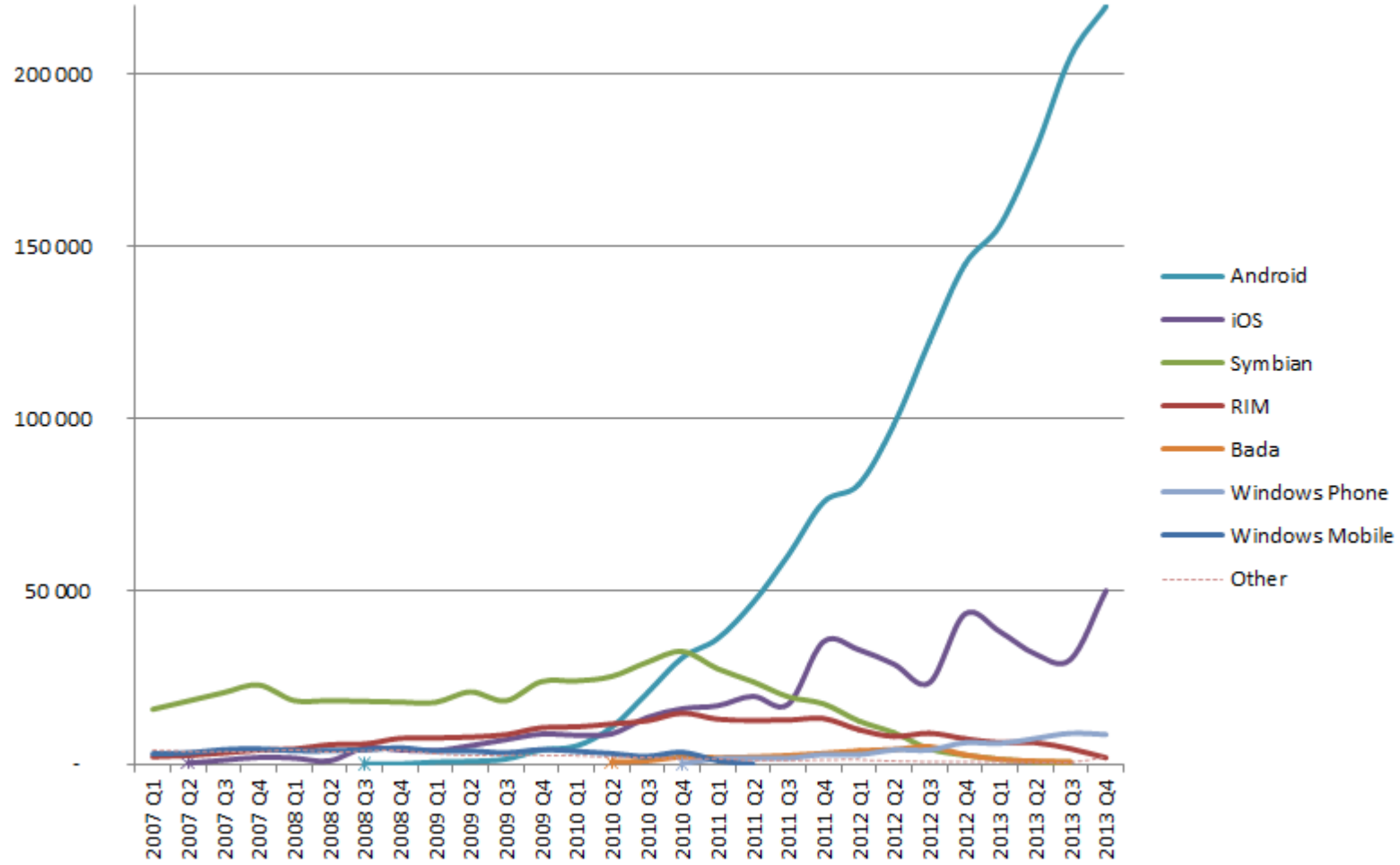
Verbreitete Devices
Smartphones

IFIXIT

DLR

Wie groß ist „Big“?

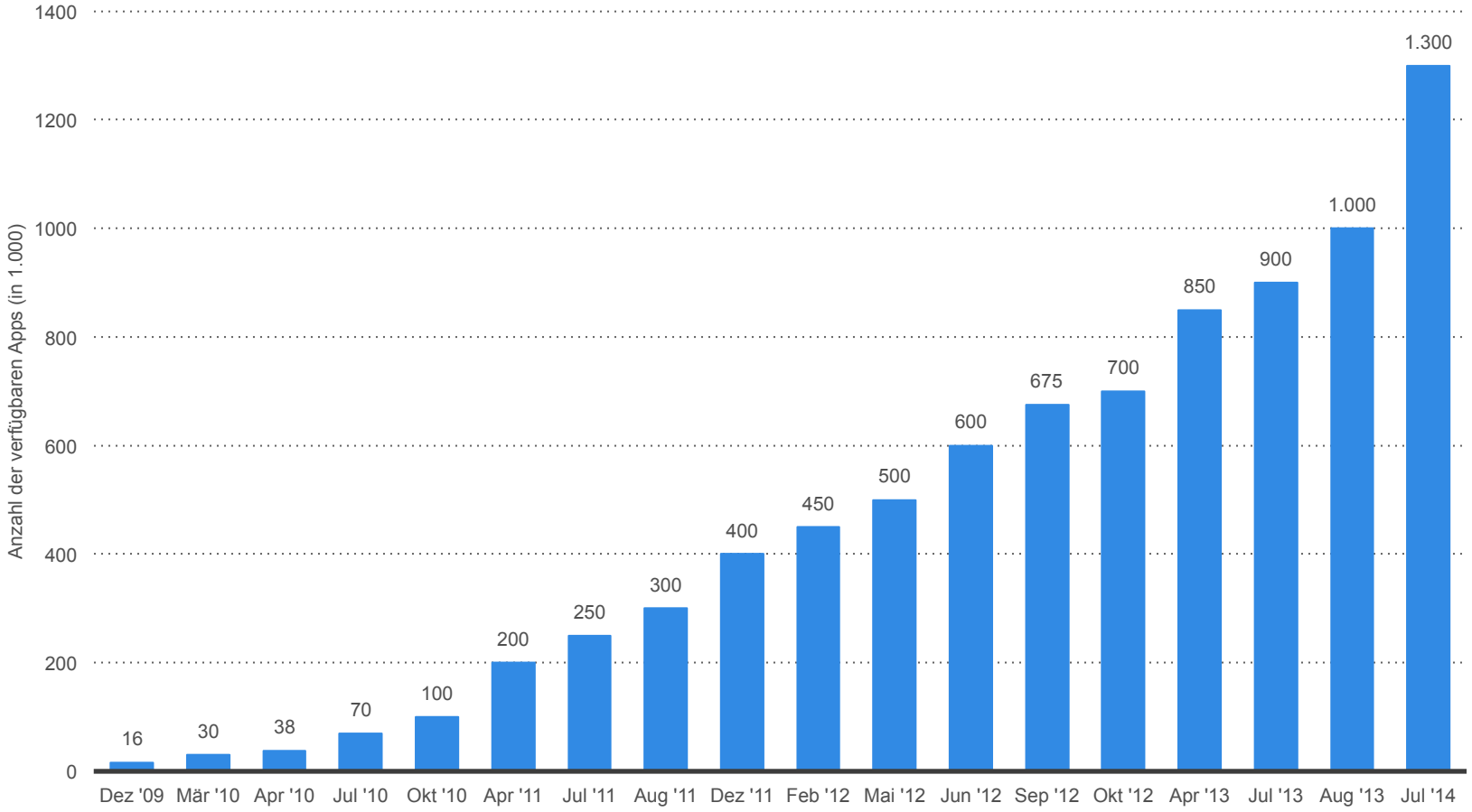
Weltweit verkaufte Smartphones



Quelle: http://en.wikipedia.org/wiki/Smartphone

Wie groß ist „Big“?

Verfügbare Apps im Google Play Store



Weltweit; Dezember 2009 bis Juli 2014, Quelle: statista GmbH, http://de.statista.com/statistik/daten/studie/74368/umfrage/anzahl-der-verfuegbaren-apps-im-google-play-store/

DLR

Sehr viele Apps…

… aber die allerwenigsten sind in Python entwickelt!

Python auf Smartphones

Frühe Technologien

- PyS60 for Symbian
- Python CE for Windows Mobile

Aktuelle Technologien

- Scripting Layer for Android (SL4A)
- Python for Android (Py4A)
- PySide / Qt for Android
- WinRT / IronPython for Windows 8
- Kivy…

Kivy
Plattformübergreifendes Python-Framework

Plattformen

- Android
- iOS
- Meego
- Windows
- Linux
- OS X
- Raspberry Pi

kivy.org

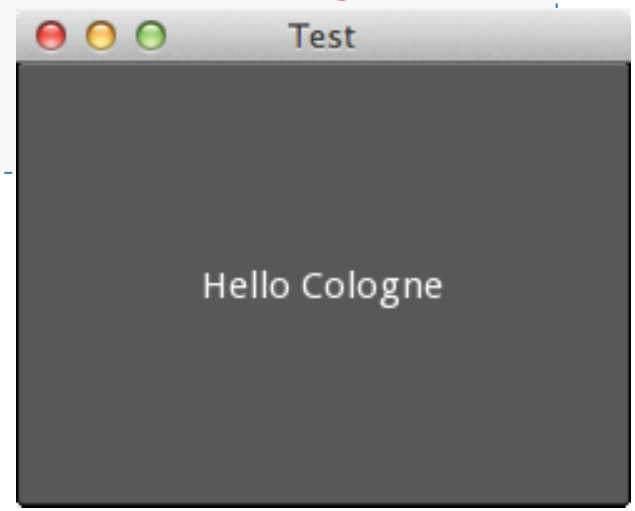
Entwicklung in Python auf allen Plattformen – keine Emulation!

Kivy
„Hello World“

```
from kivy.app import App
from kivy.uix.button import Button

class TestApp(App):
    def build(self):
        return Button(text='Hello Cologne')

TestApp().run()
```

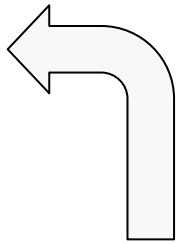


Kivy
Sprache „KV“ für Layout und Grafik

```
from kivy.app import App

class HelloApp(App):
    pass

HelloApp().run()
```



Datei **hello.kv**
definiert Root-Widget

```
#:kivy 1.0

Button:
    text: ‘Hello Hamburg’
```

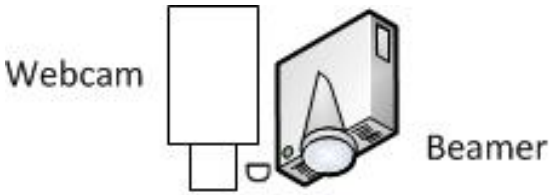
Kivy Apps
Verfügbar zum Beispiel im Google Play Store

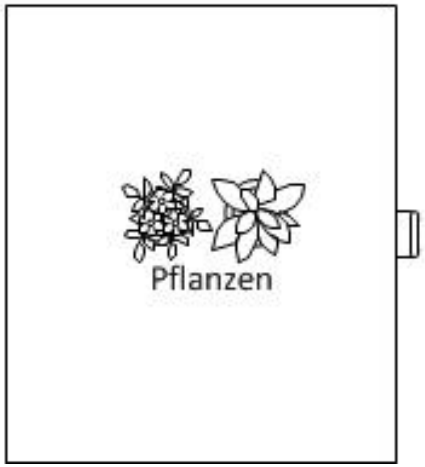
Kivy Apps
Geeignet für Prototypen

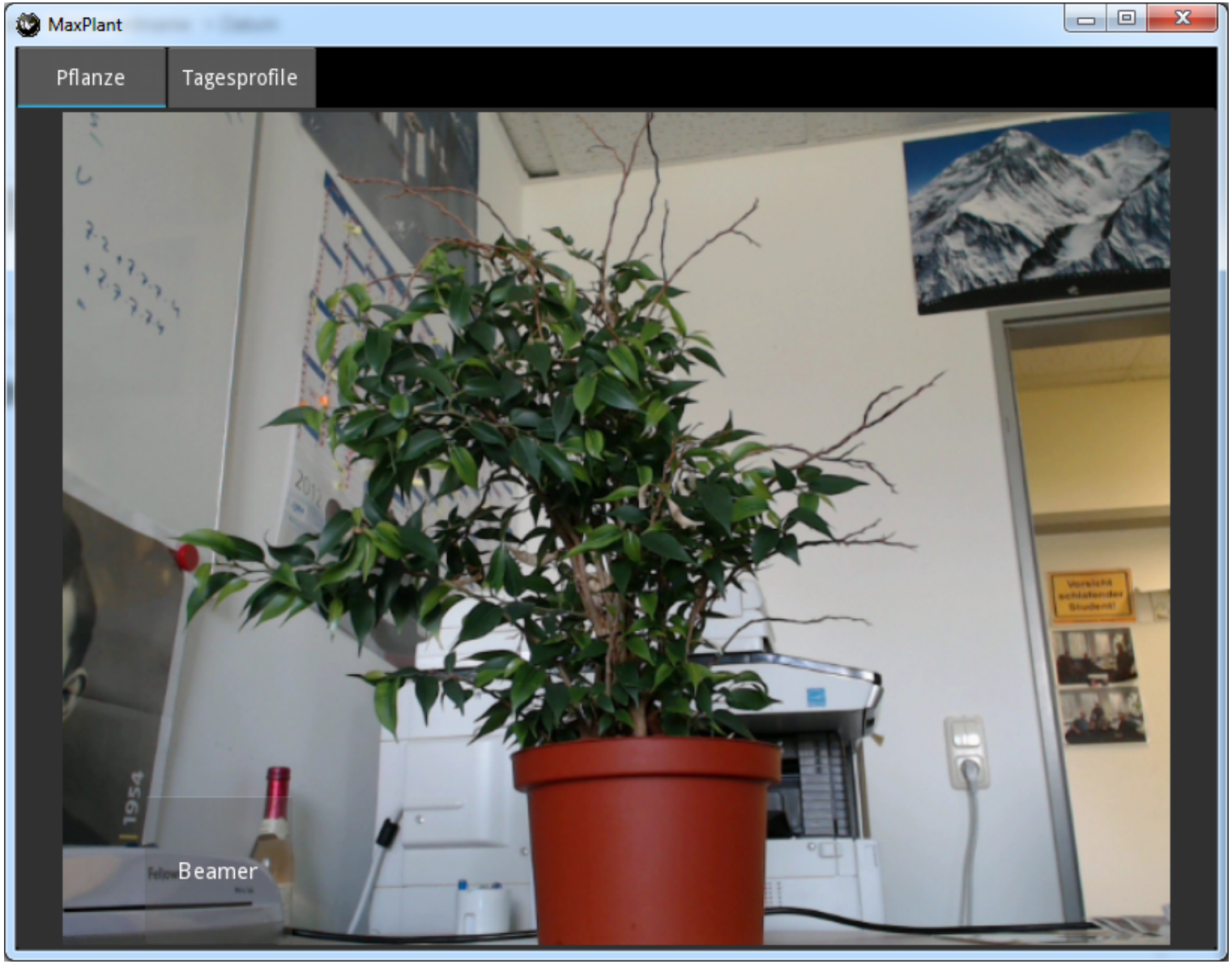
Kivy zur Erstellung von GUIs und Apps
Beispiel aus der Raumfahrtbiologie

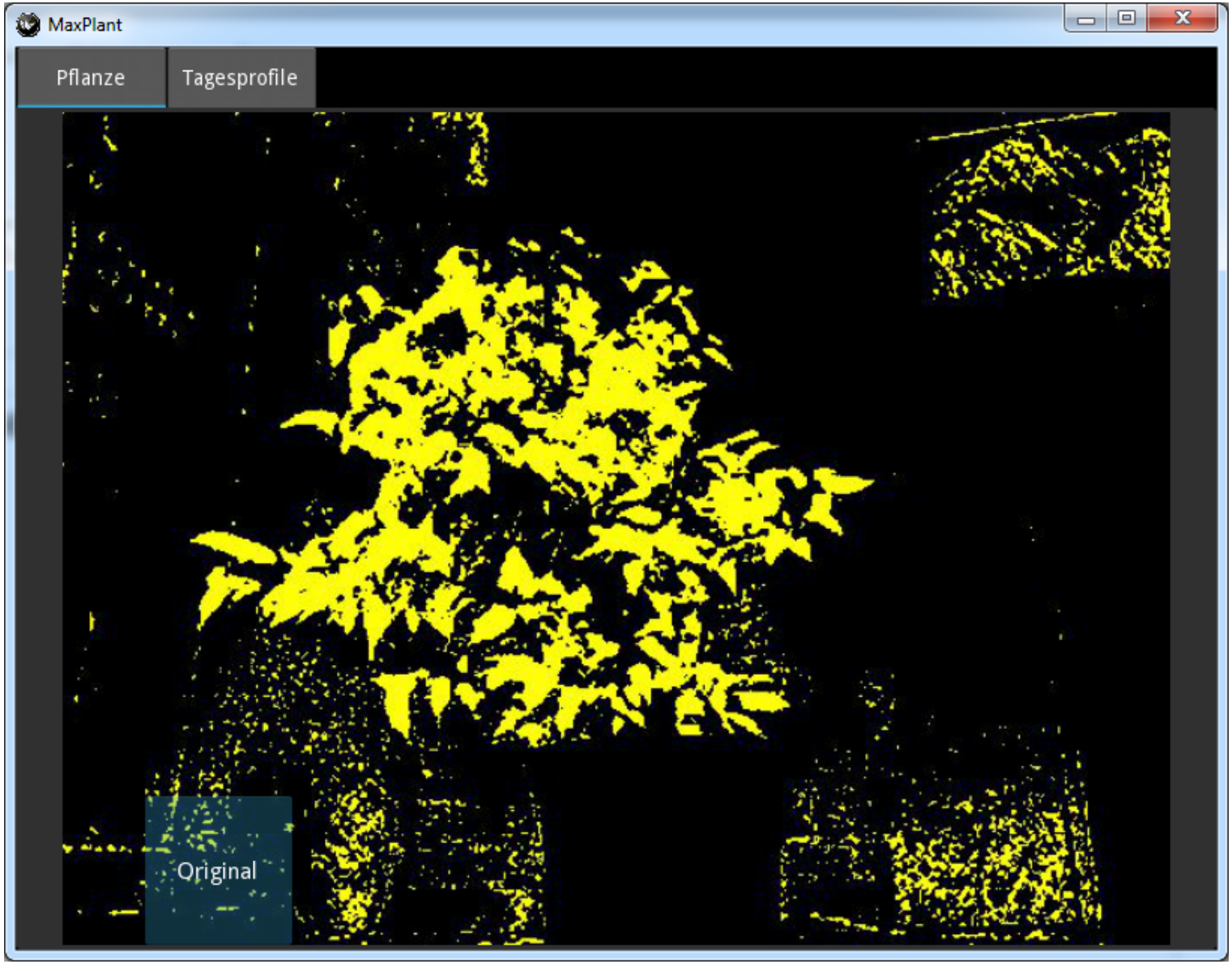
Pflanzenbeleuchtung

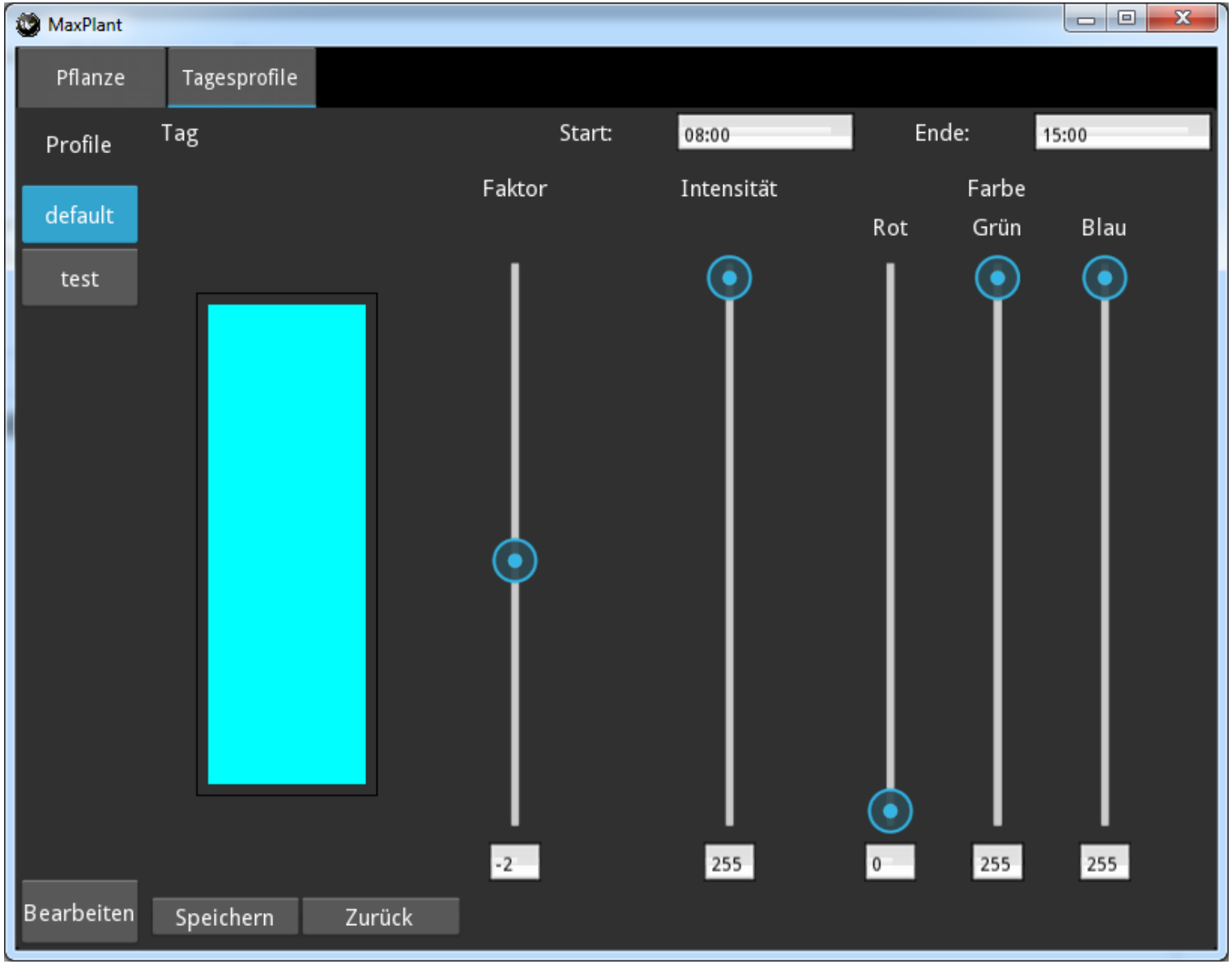
- Webcam nimmt Bild auf
- Rechner erkennt Pflanze
- Rechner berechnet anhand von Einstellungen ein Ausgabebild
- Lichtquelle (z.B. Beamer) beleuchtet die Pflanze mit dem Bild





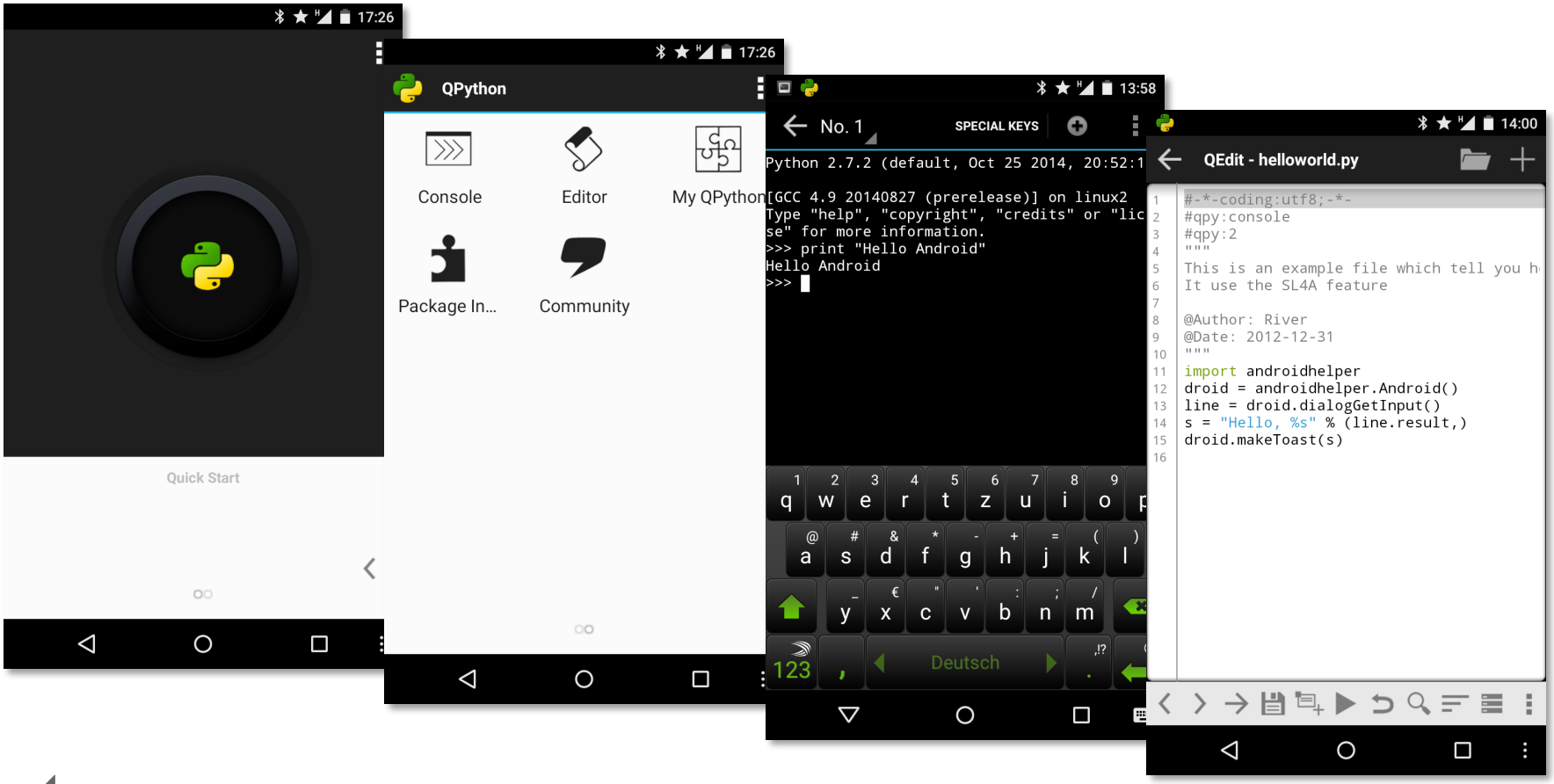

MaxPlant
Pflanze
Tagesprofile
Beamer


MaxPlant
Pflanze
Tagesprofile
Original


MaxPlant
Pflanze
Tagesprofile
Profile
Tag
Start:
08:00
Ende:
15:00
default
test
Faktor
Intensität
Farbe
Rot
Grün
Blau
-2
255
0
255
255
Bearbeiten
Speichern
Zurück

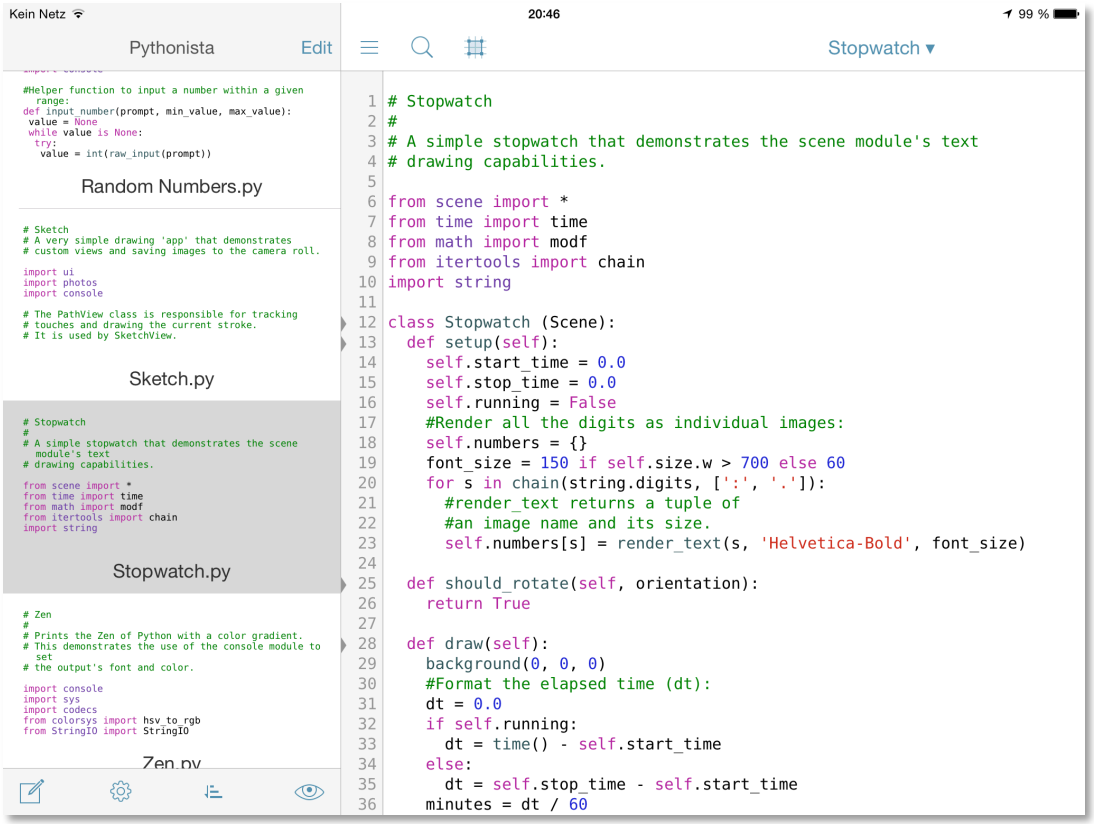
Python auf Smartphones
Weitere Möglichkeiten…

QPython – Python on Android (`http://qpython.com`)



Python auf Smartphones
Weitere Möglichkeiten…

Pythonista – Python on iOS
`http://omz-software.com/pythonista`



Big Computers

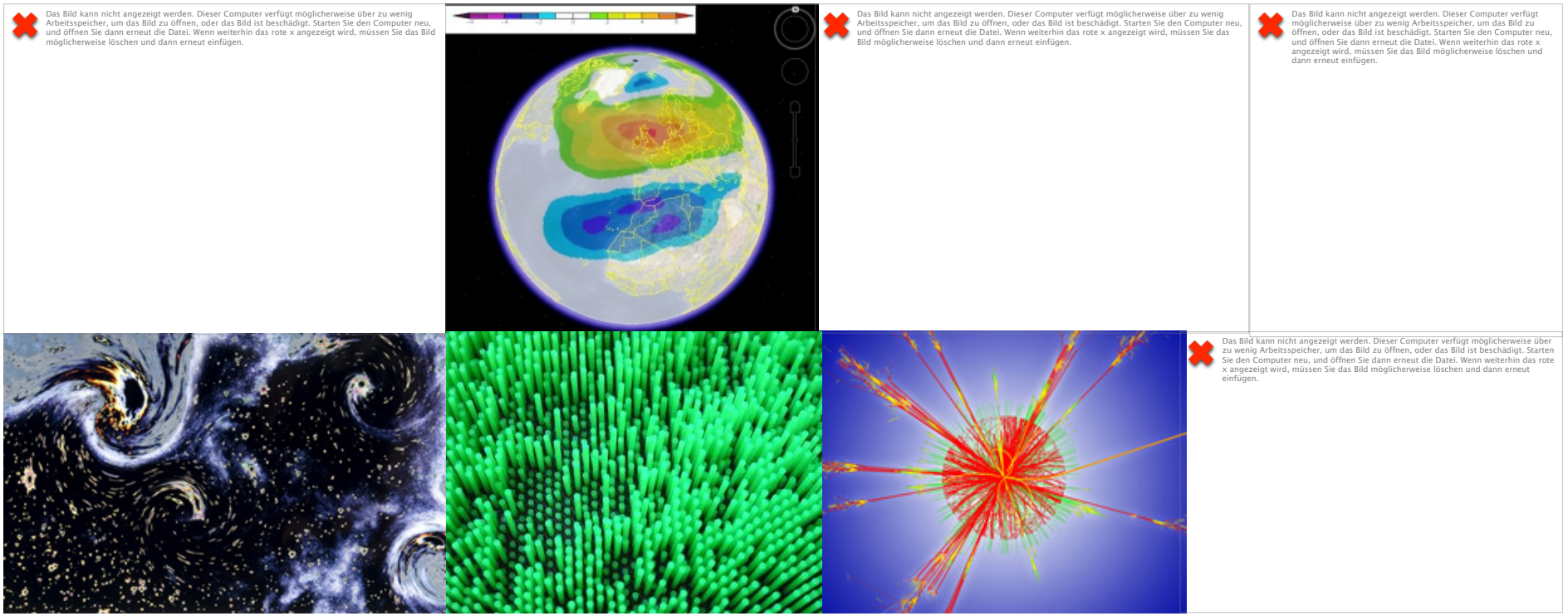
High Performance Computing

DLR

High Performance Computing (HPC)
Spezialgebiet des Wissenschaftlichen Rechnens

Wenn der Arbeitsplatzrechner nicht mehr ausreicht

- Hohe Rechenleistung und hoher Speicherbedarf



Bilder: http://www.isgtw.org

Wie groß ist „Big“?
Aktuelle Supercomputer

Aktuelle Supercomputer haben

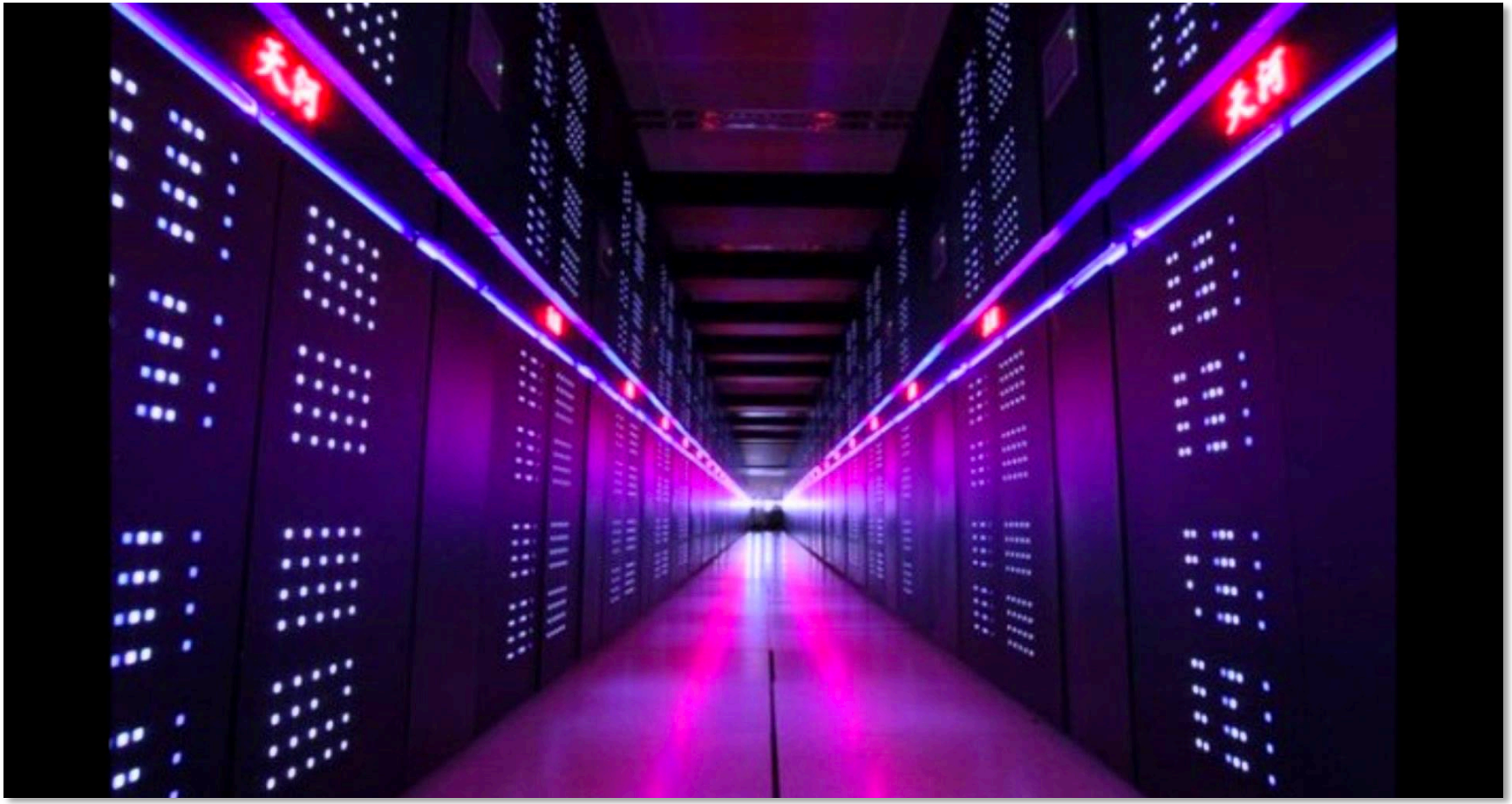
> 1.000.000 Cores

> 10 PetaFLOPS

Die drei derzeit größten Systeme (Nov 2014)

1. **Tianhe-2** (Guangzhou, China)
   3.120.000 Cores, 33,8 PetaFLOPS
2. **Titan** (ORNL, USA)
   560.640 Cores, 17,5 PetaFLOPS
3. **Sequoia** (LLNL, USA)
   1.572.864 Cores, 17,1 PetaFLOPS

Supercomputer
Tianhe-2 (天河二号)



Supercomputer
Titan

Quelle: https://www.olcf.ornl.gov/titan/

Supercomputer
Sequoia

Quelle: https://asc.llnl.gov/computing_resources/sequoia/

High Performance Computing
Programmiertechnologien

MPI (Message Passing Interface)

- API für Distributed-Memory-Architekturen in C, C++, Fortran, ...

OpenMP (Open Multi-Processing)

- API für Shared-Memory-Architekturen in C, C++, Fortran

OpenACC (Open Accelerators)

- API für heterogene CPU/GPU-Systeme in C, C++, Fortran

Global Arrays Toolkit

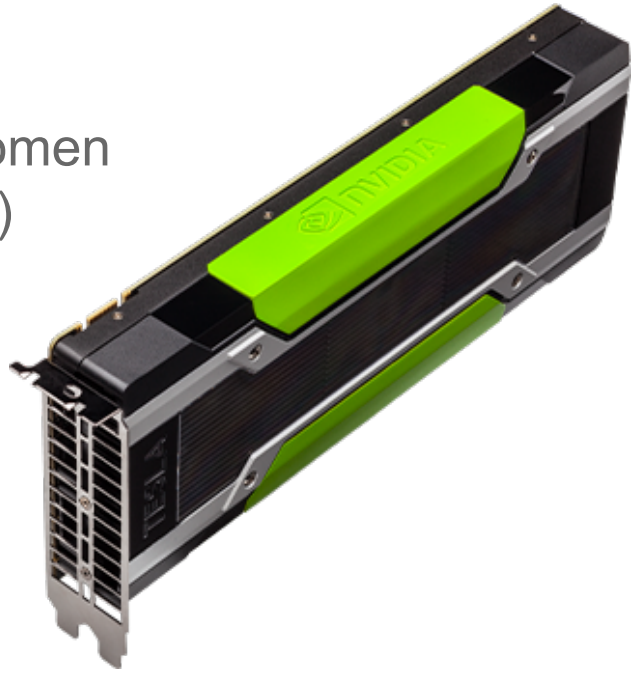
- API für Shared-Memory-Programmierung auf Distributed-Memory-Architekturen in C, C++, Fortran und Python

GPGPU

General-purpose computing on graphics processing units

Architektur

- Viele Core pro Node
- Geeignet für Prozessierung von Datenströmen (viele parallele unabhängige Datenpunkte)



Programmiertechnologien

- CUDA
  - API von NVIDIA in C
  - Python-Binding: PyCUDA
- OpenCL
  - Offenes Framework für heterogene Systeme
  - Python-Binding: PyOpenCL

Bild: NVIDIA

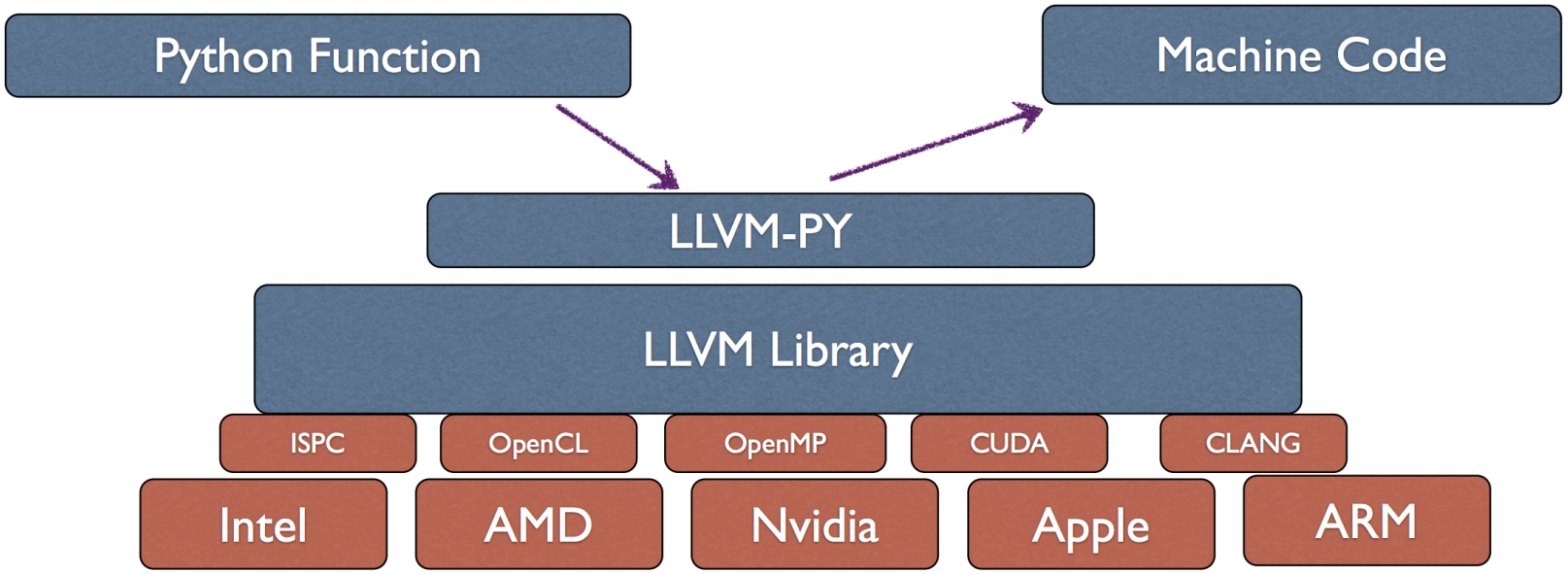
Numba

Optimierungs-Compiler für Python

CONTINUUM ANALYTICS

Just-in-time-Compiler

- Annotationen
- Nutzt LLVM

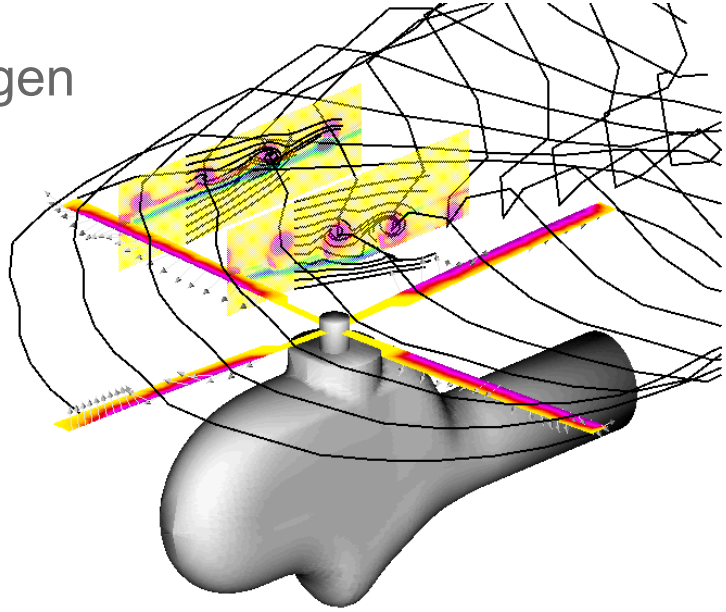


DLR

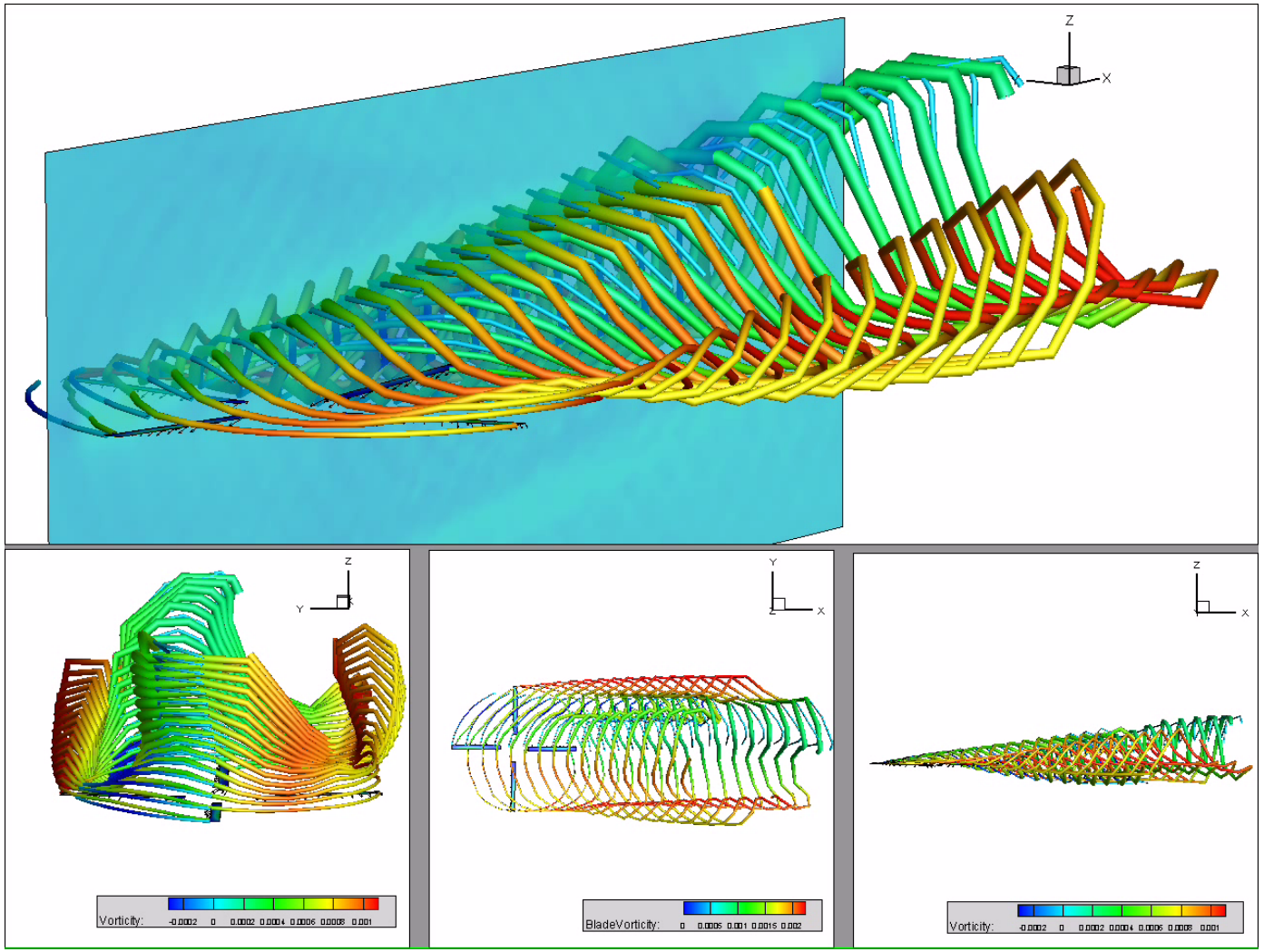
Beispiel Free-Wake
Simulation von Hubschrauber-Rotoren

Free-Wake (DLR)

- Simulation dreidimensionaler Strömungen um einen aktiv gesteuerten Rotor eines Helikopters
- Code entwickelt 1994-1996
    - MPI-parallelisiert in Fortran
- Aufwendige Performance-Optimierung 2013-2014
    - MPI und Open ACC



Visualisierung der Wirbel



Kern-Schleifen von Free-Wake (Standard Python)

```
for iblades in range(numberOfBlades):
    for iradial in range(1, dimensionInRadialDirection):
        for iazimutal in range(dimensionInAzimualDirectionTotal):
            for i1 in range(len(vx[0])):
                for i2 in range(len(vx[0][0])):
                    for i3 in range(len(vx[0][0][0])):
                        # wilin-Aufruf 1
for iblades in range(numberOfBlades):
    for iradial in range(dimensionInRadialDirection):
        for iazimutal in range(1, dimensionInAzimualDirectionTotal):
            for i1 in range(len(vx[0])):
                for i2 in range(len(vx[0][0])):
                    for i3 in range(len(vx[0][0][0])):
                        # wilin-Aufruf 2
for iDir in range(3):
    for i in range(numberOfBlades):
        for j in range(dimensionInRadialDirection):
            for k in range(dimensionInAzimualDirectionTotal):
                x[iDir][i][j][k] = x[iDir][i][j][k] +
                                   dt * vx[iDir][i][j][k]
```

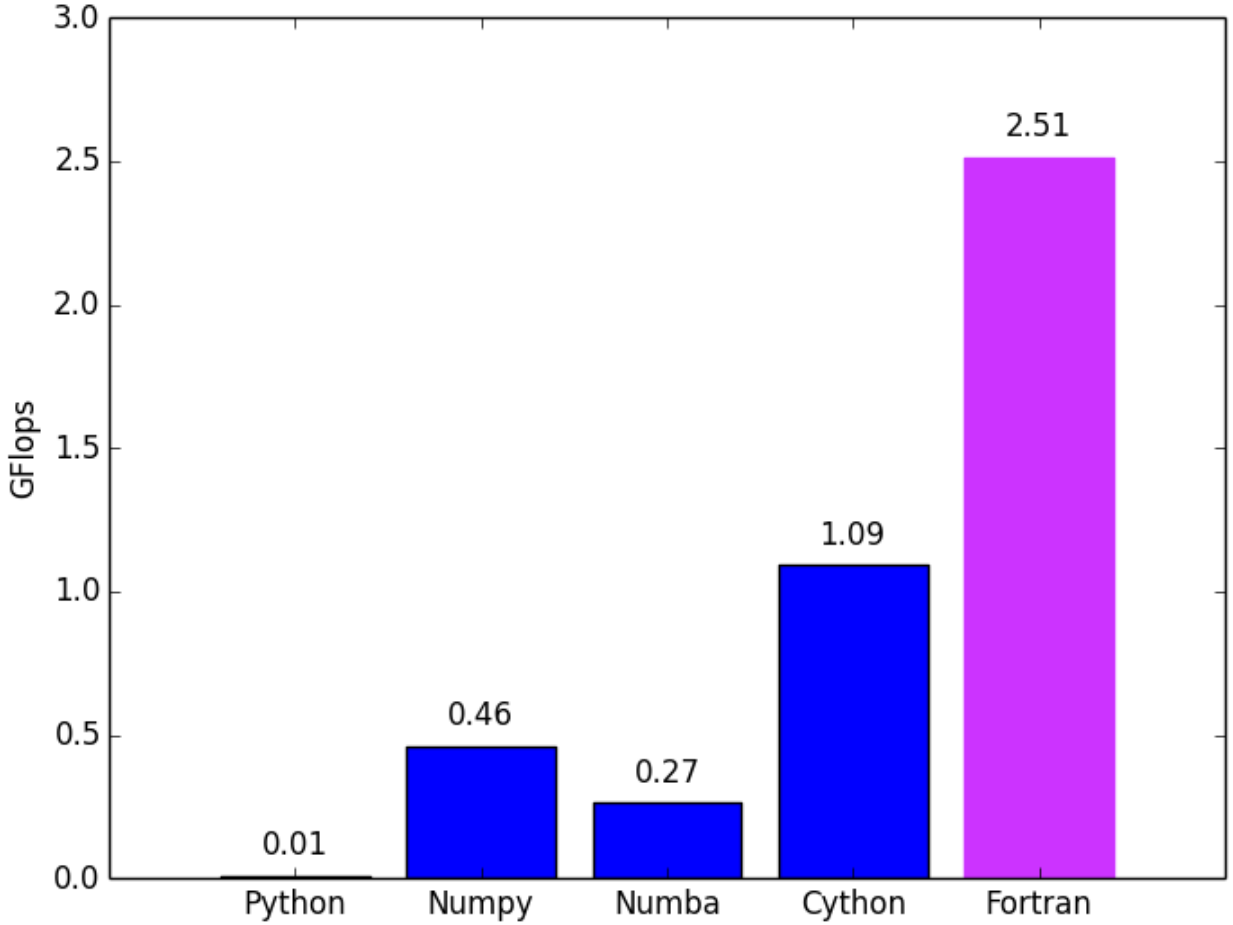
Free-Wake
Performance-Vergleich Fortran – Python

Vergleich der hoch-optimierten Fortran-Version mit parallelen Python-Versionen

- Multi-core CPUs
  - Cython mit OpenMP
  - Python-Bindings für Global Array Toolkit
- GPGPUs
  - NumbaPro

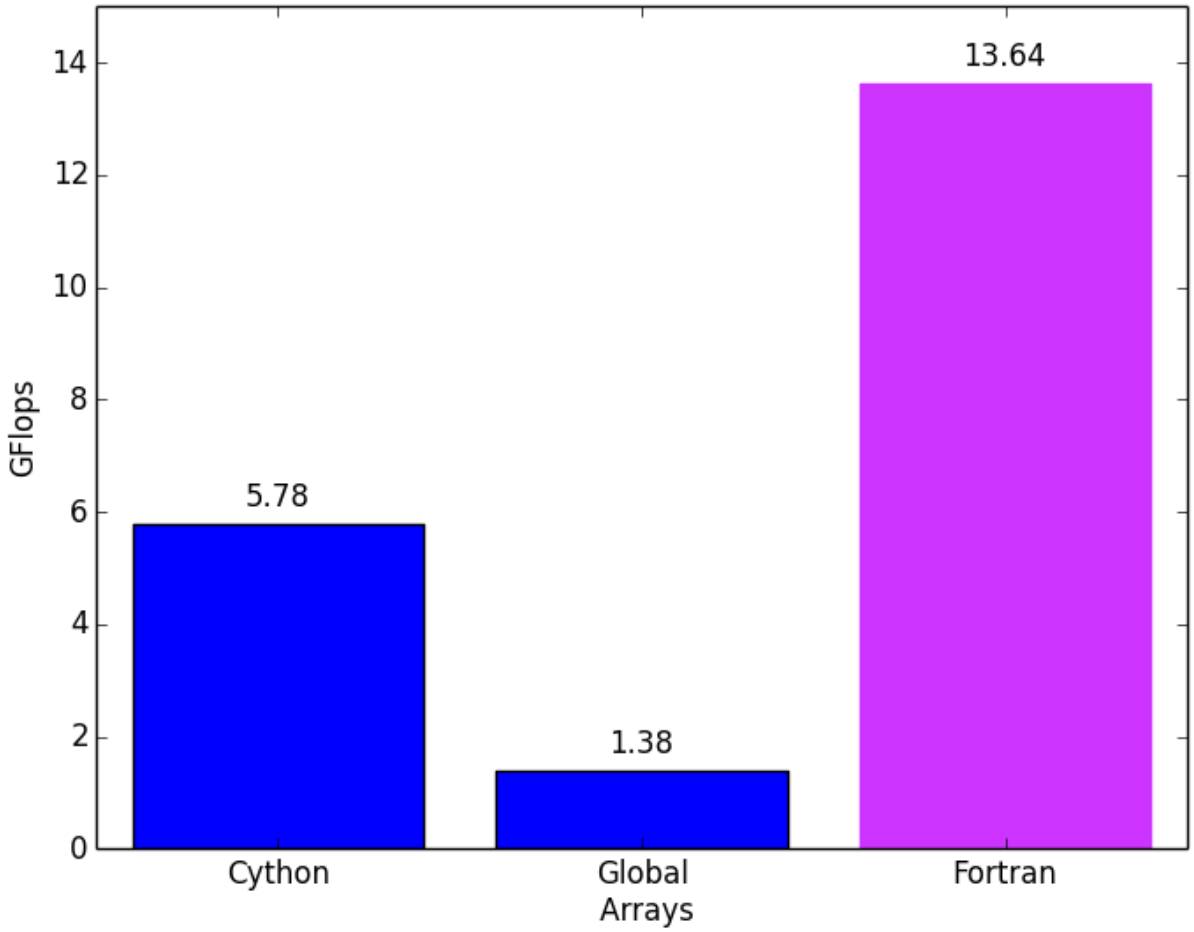
Performance-Tests

Single-Core Performance (Xeon E5645, 6 Cores)

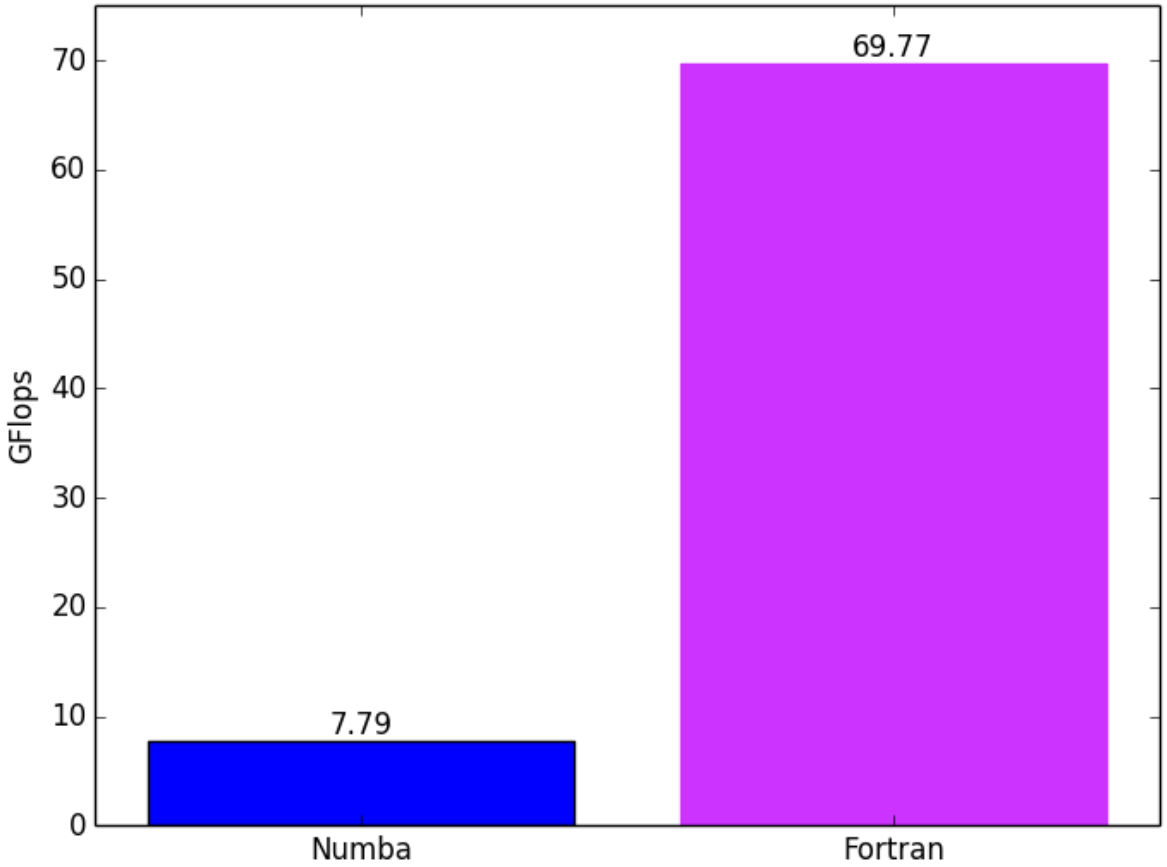


Performance-Tests

Multi-Core Performance (Xeon E5645, 6 Cores)



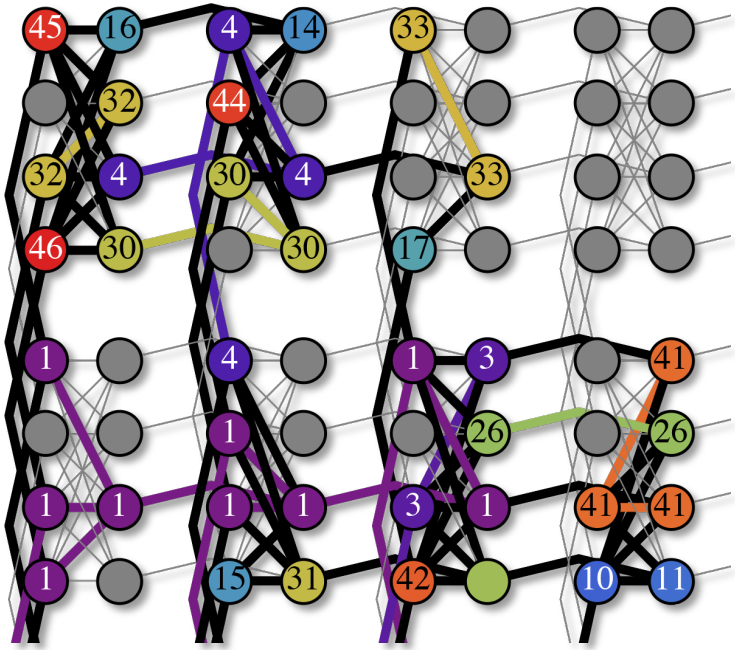
Performance-Tests
GPGPU Perf. (NVIDIA Tesla C2075, 448 CUDA-Cores)



Neue Rechnerarchitekturen
Quantencomputer

- Adiabatische Quantencomputer

$$H = \sum_i h_i Z_i + \sum_{i<j} J^{ij} Z_i Z_i + \sum_{i<j} K^{ij} X_i X_i$$



Bilder: NASA

Big Applications

„Killer“-Applikationen in Wissenschaft und Technologie

DLR

Wichtige Anwendungssoftware
Wissenschaft und Technik

Viele kommerzielle Anwendungen sind noch Standard

- Microsoft Excel
- MATLAB
- IDL
- Fortran-Compiler

Der Weg nach Python...

- Open Source
- Einheitliche Sprache für viele Anwendungsgebiete

Microsoft Excel
Tabellenkalkulation

Wesentliche Funktionen

- Tabellen
- Sortier-, Gruppier-, Filterfunktionen
- Pivot-Tabellen
- Diagramme

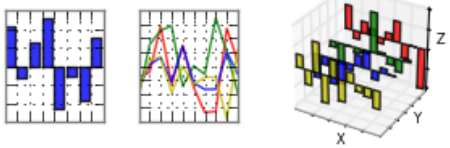
Python-Alternative

- IPython
- pandas

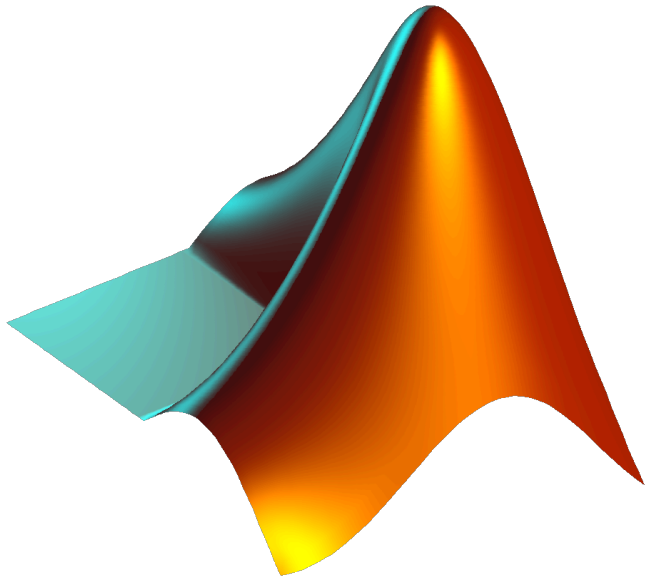
IP[y]: IPython
Interactive Computing

pandas

$$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$$



The MathWorks MATLAB
Numerische Matrixberechnungen



Wesentliche Funktionen

- Eigene proprietäre Programmiersprache
- Viele Anwendungs-Toolboxes
  z.B. Statistik, Signal- und Bildverarbeitung

Python-Alternative

- NumPy
- Matplotlib

Nützliche Quelle: `wiki.scipy.org/NumPy_for_Matlab_Users`

IDL – Interactive Data Language
Analyse und Visualisierung von Daten

Wesentliche Funktionen

- Array-basierte Programmiersprache
- Gute Bildverarbeitungsfunktionen

IDL

Python-Alternative

- IDL-nach-Python-Compiler PIKE

Tørsion Analytics

DLR

PIKE
Beispiel-Codes

Torsion Analytics

```
;; Simple image/ plotting and graphics tests

pro MRI_demo

  ;Load demo data file
  file = filepath("mri500x300x5.dat", subdirectory=["data"])
  print,file

  device, decomposed=0
  loadct, 3

  openr, lun, file, /get_lun

  ;Associate a variable with a data file
  img = assoc(lun, bytarr(500, 300))

  !P.multi=[0,0,0,0]
  window, 0, xsize=500, ysize=300,
            title='MRI Demo - Flicker Loop'

  ;Display the five images in a loop
  for j=0, 2 do begin
    for i=0, 4 do begin
      tvscl, img[i]
      wait, 0.1
    endfor
  endfor
. . .
```

DLR

PIKE
Beispiel-Codes

Torsion Analytics

```
import numpy as np
import pike

def mri_demo( ):

#Load demo data file
    pike.setArrayOrder("0and1") # expected array ordering

    # %;   Define detected undefined variables
    lun = 0

    file = pike.filepath("mri500x300x5.dat”, subdirectory=pike.catarr(["data"]))
    pike.print_(file)

    pike.device(decomposed=0)
    pike.loadct(3)

    lun = pike.openr(lun, file, get_lun=True)

    #Associate a variable with a data file
    img = pike.assoc(lun, pike.bytarr(500, 300))

    pike.sysv.P.multi = pike.catarr([0, 0, 0, 0])
    pike.window(0, title='MRI Demo - Flicker Loop', xsize=500, ysize=300)

    #Display the five images in a loop
    for j in xrange(np.int16(0), (np.int16(2))+(1)):
        for i in xrange(np.int16(0), (np.int16(4))+(1)):
            pike.tvscl(img[i])
            pike.wait(0.1)
. . .
```

DLR

PIKE
Beispiel-Codes

Torsion Analytics

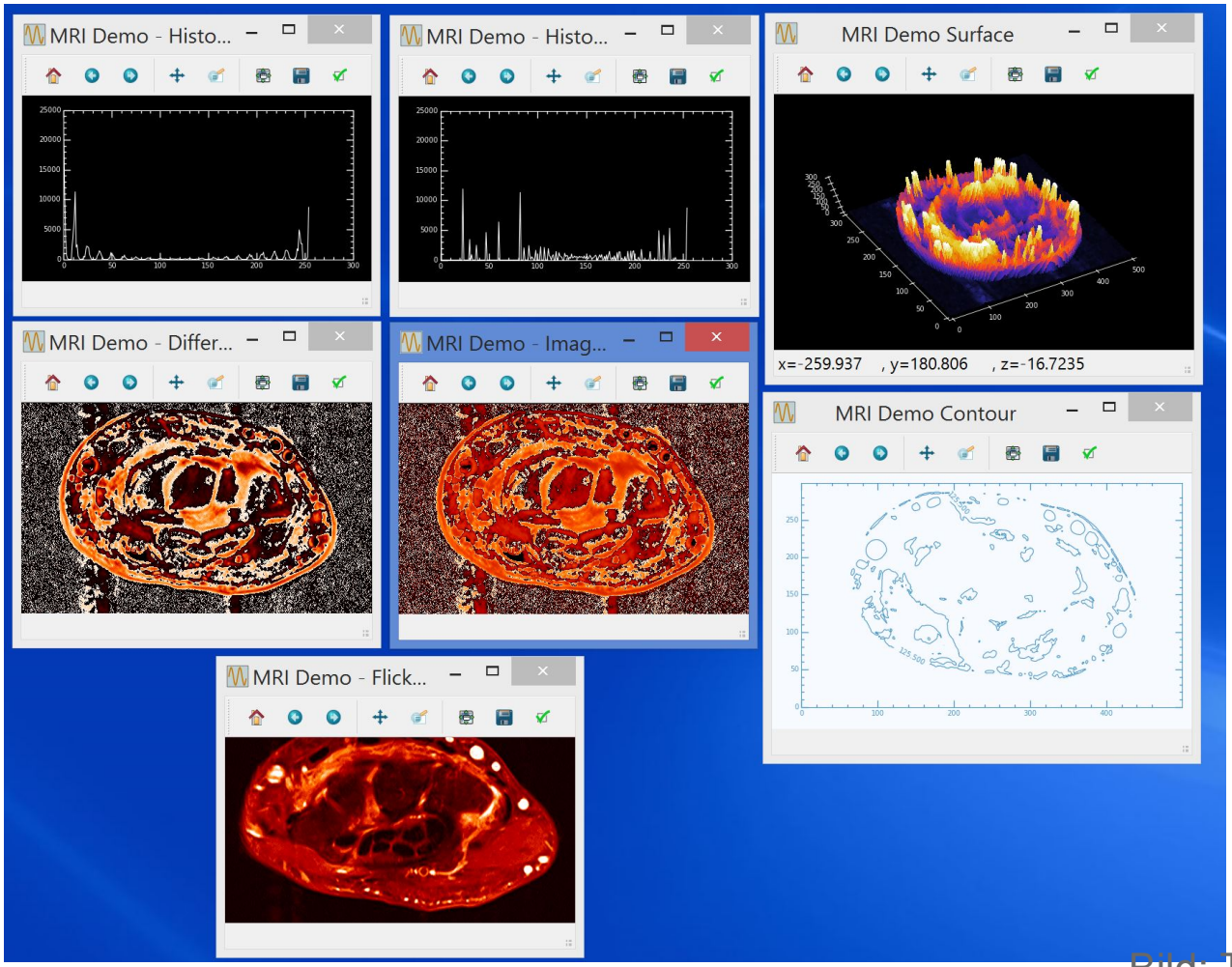
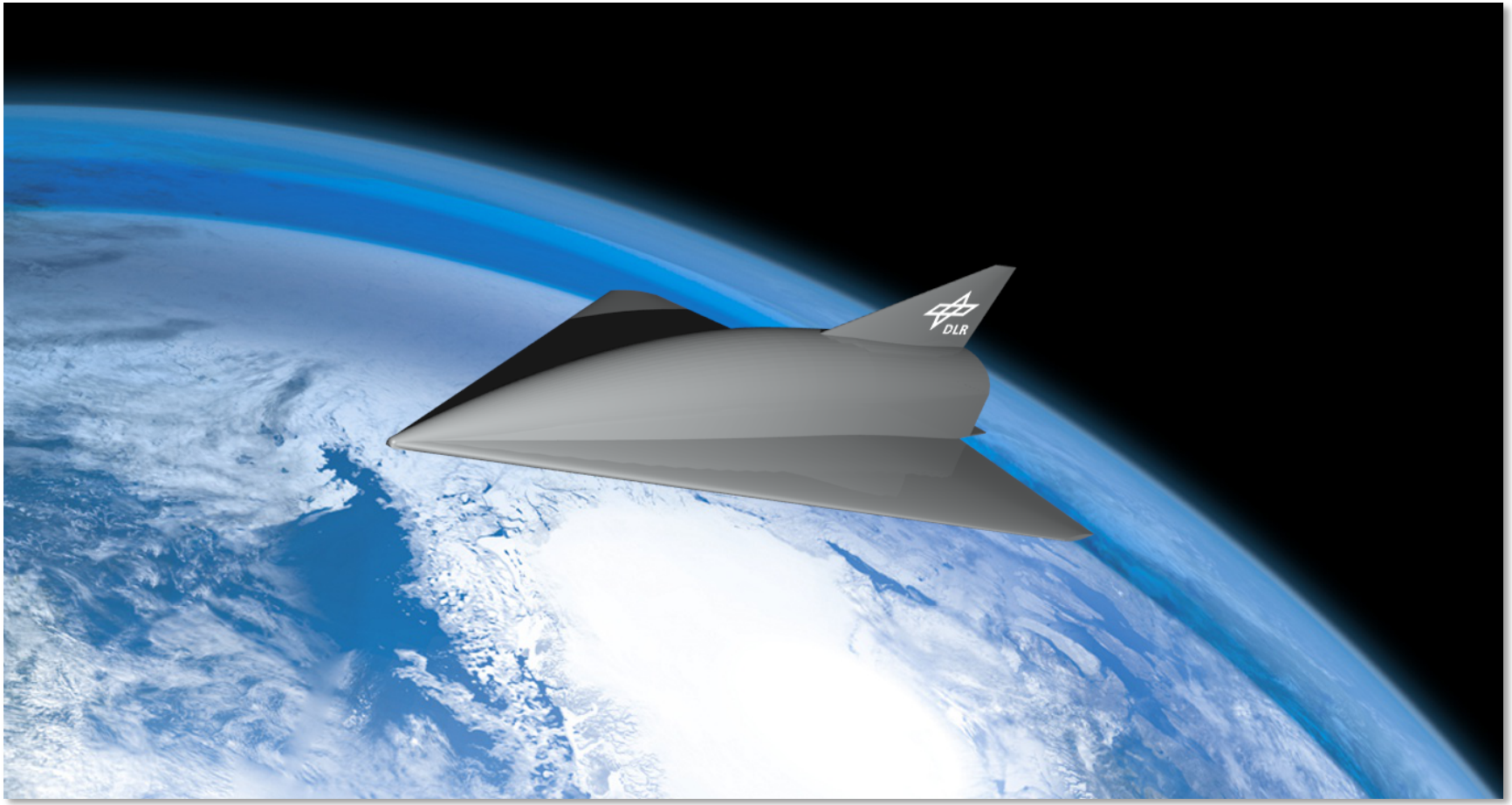


Bild: Torsion Analytics

DLR

Entwerfen von Raumfahrzeugen

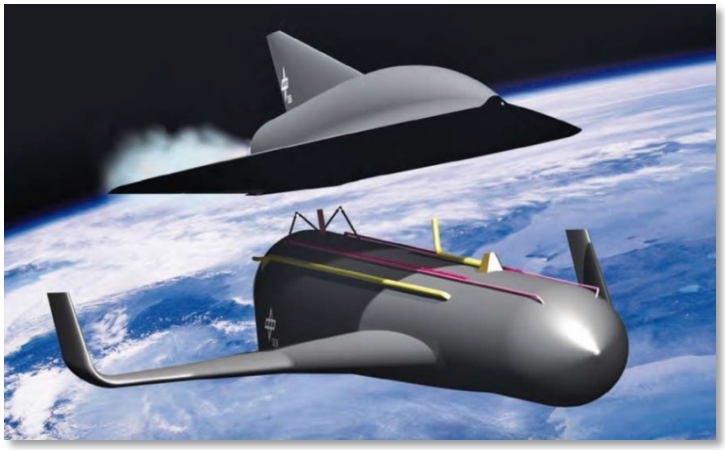


DLR

Entwerfen von Raumfahrzeugen
Beispiel: Der DLR SpaceLiner

SpaceLiner

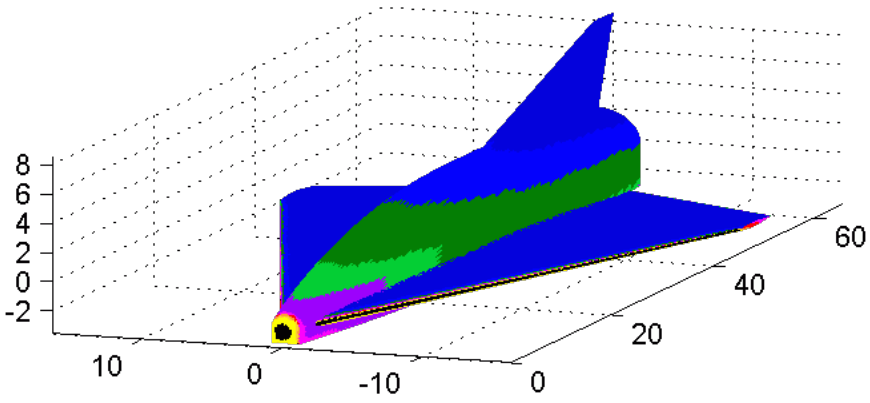
- Konzeptstudie für Passagiertransport
- Mittelding zwischen Flugzeug und Raumschiff
- Langstreckenflüge mit Hyperschallgeschwindigkeit (> Mach 5)
- Strecke Europa – Australien in 90 Min.
- Hochaufstieg mit Booster auf ca. 85 km
- Gleitflug des Orbiters mit ca. Mach 20



Simulation in der Entwurfsphase
Wärmeentwicklung beim Wiedereintritt

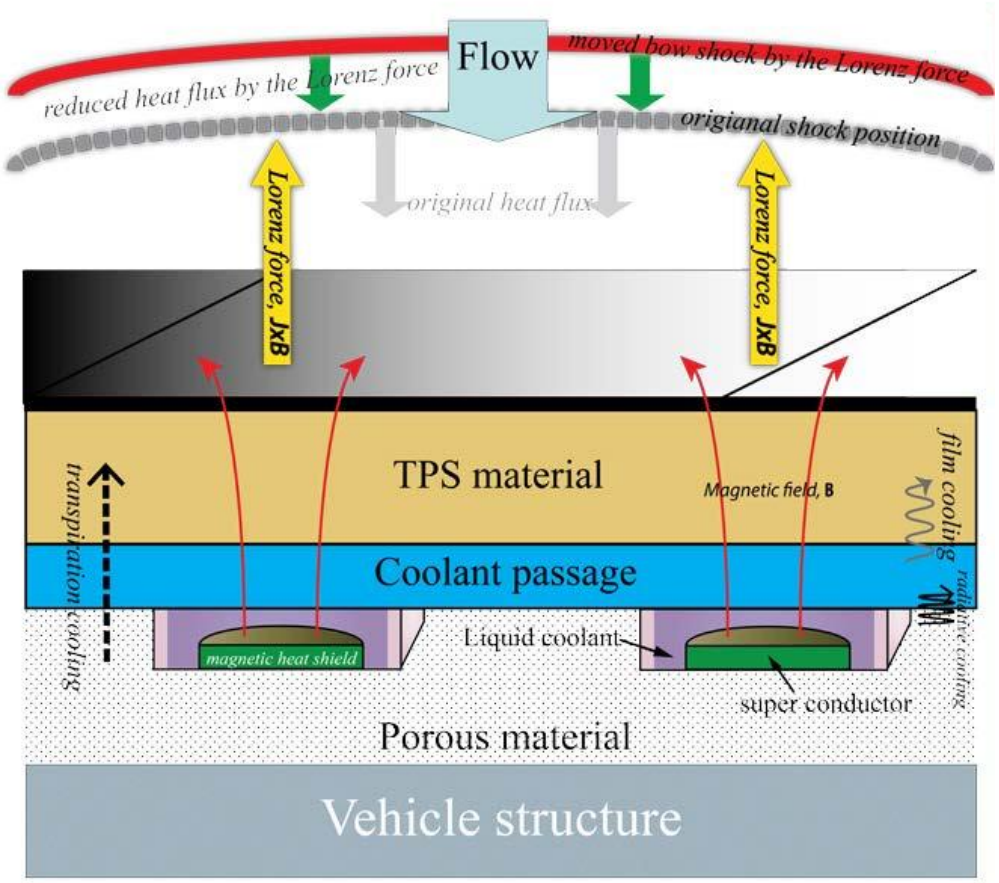
Simulation mit verschiedenen Wärmeschutzsystemen

- Wasserkühlung durch Verdampfung
- Hochwärmeleitende Faserverbundstoffe



Wärmeschutzsystem

Magnetohydrodynamik mit supraleitenden Magneten



Lorentzkraft in der Natur

Aurora Borealis

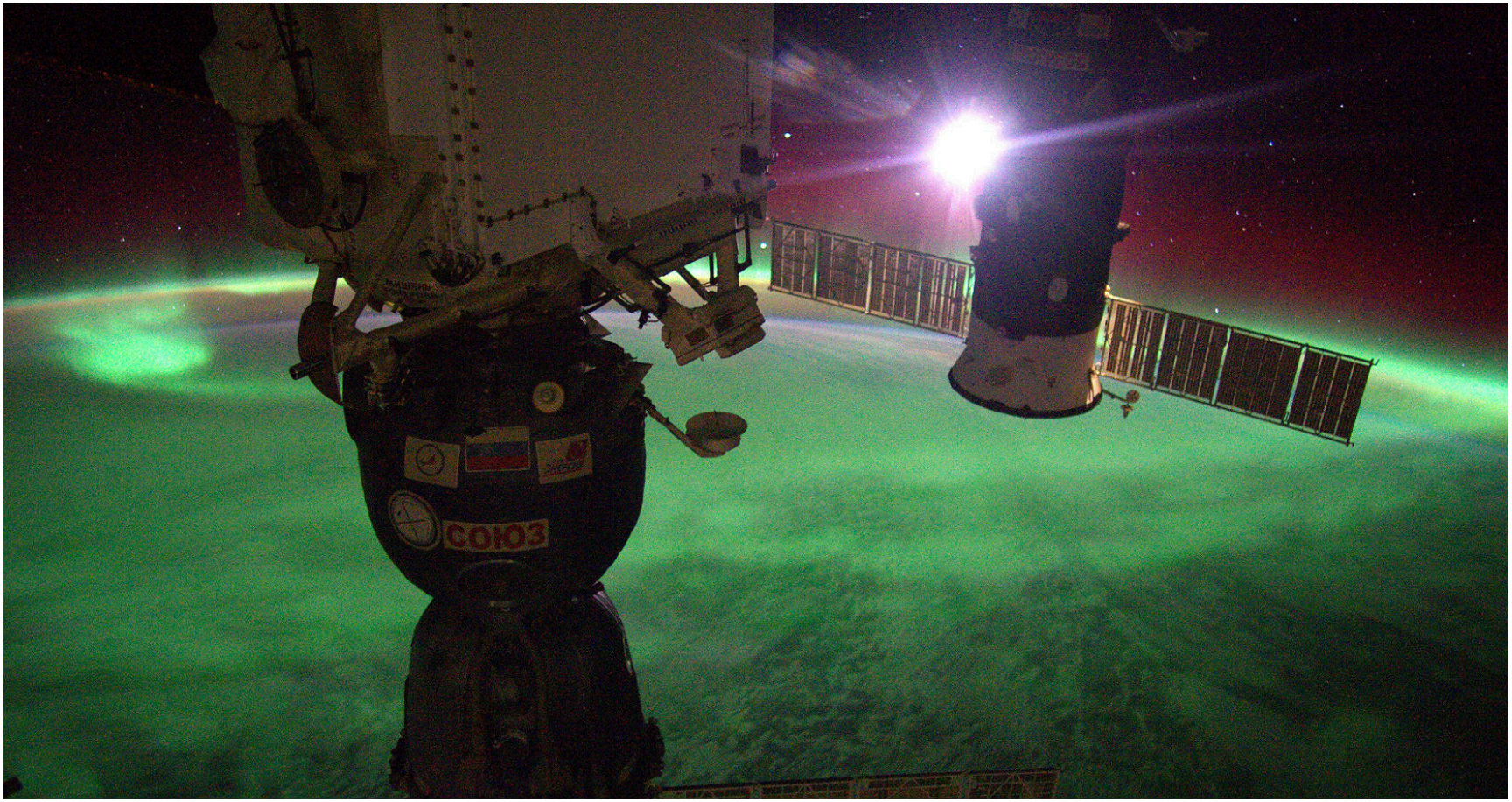
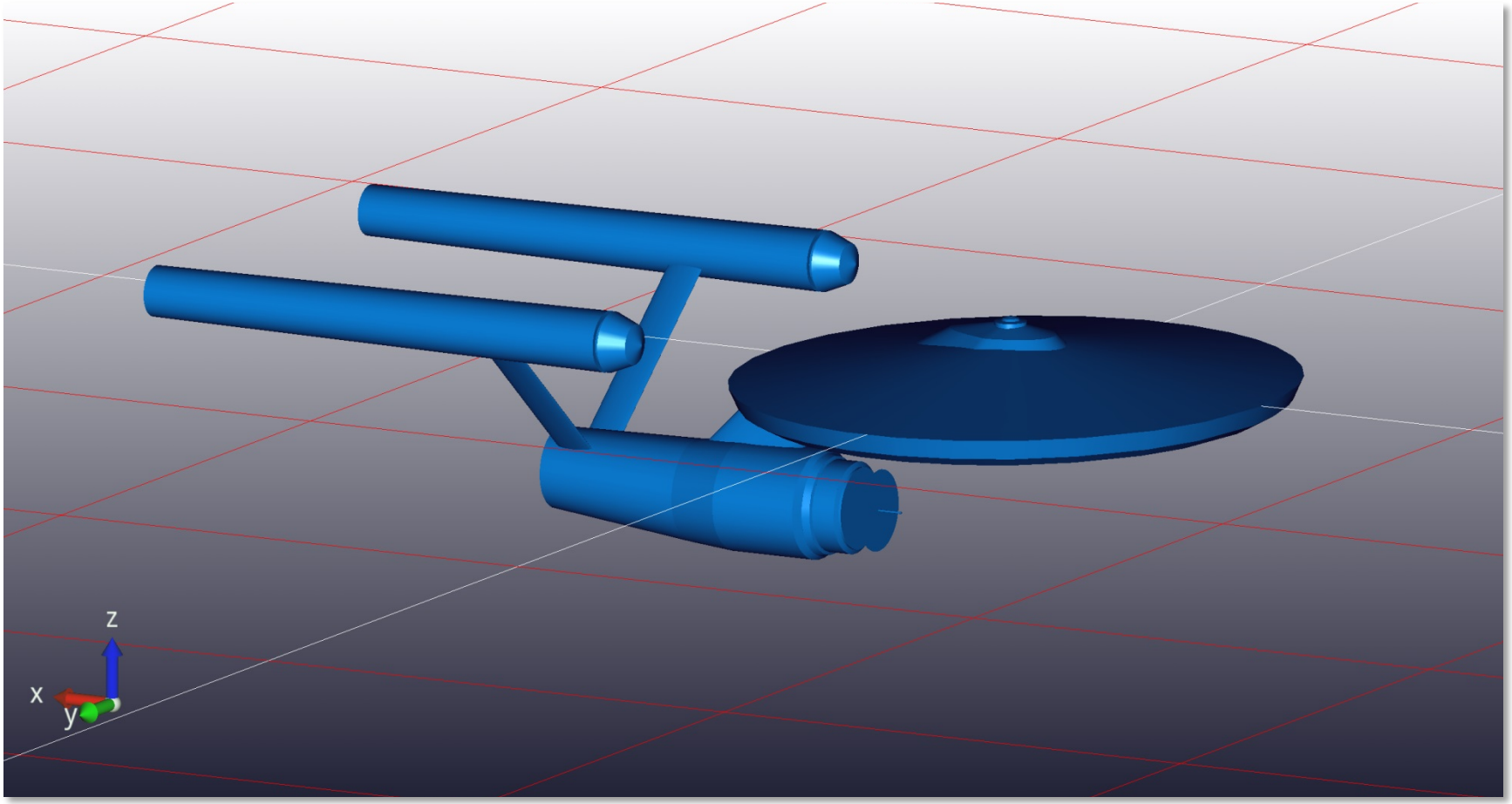


Bild: Alexander Gerst (https://twitter.com/Astro_Alex/status/507212904689848320)

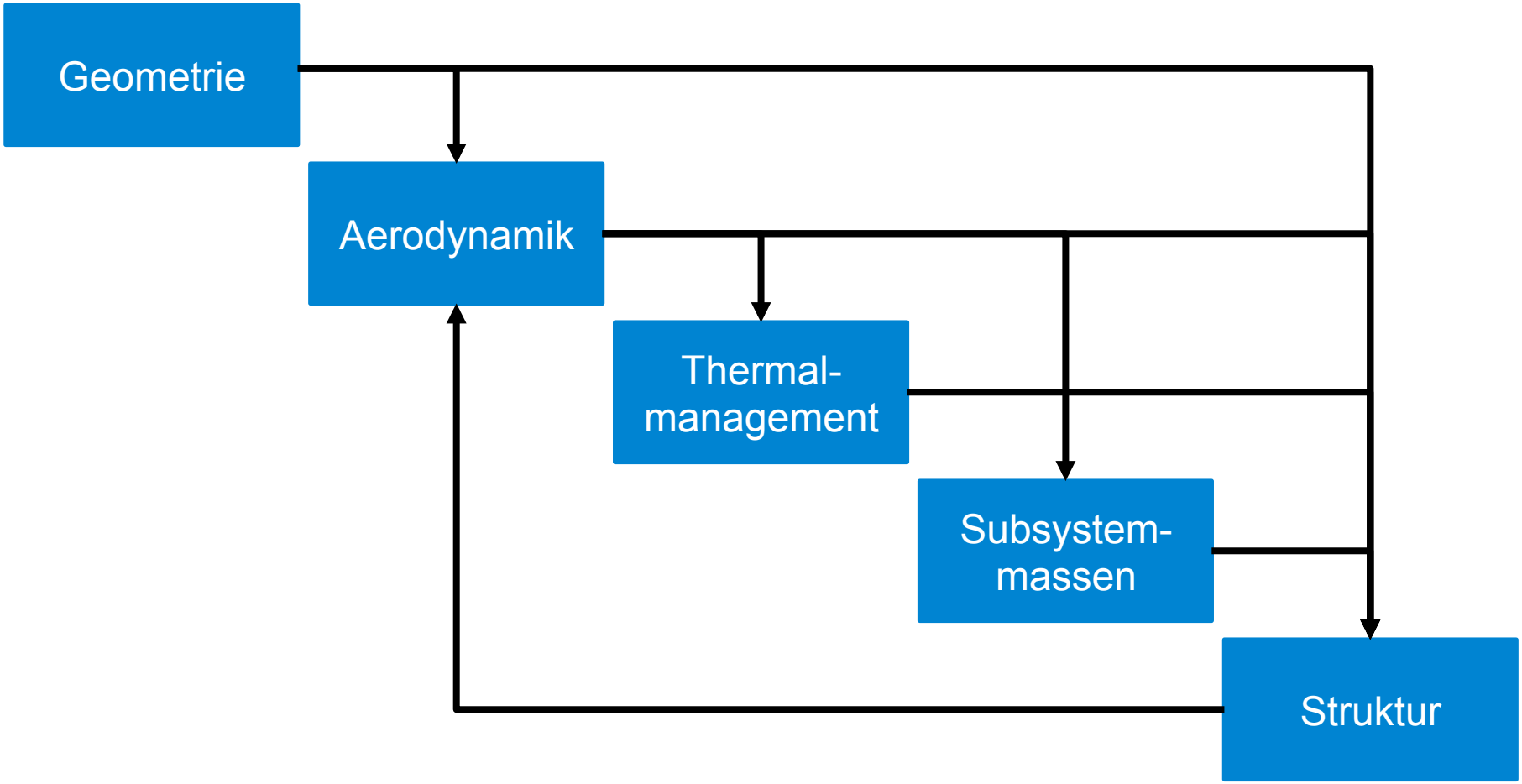
Lorentzkraft in der Raumfahrt

Schutzschilde

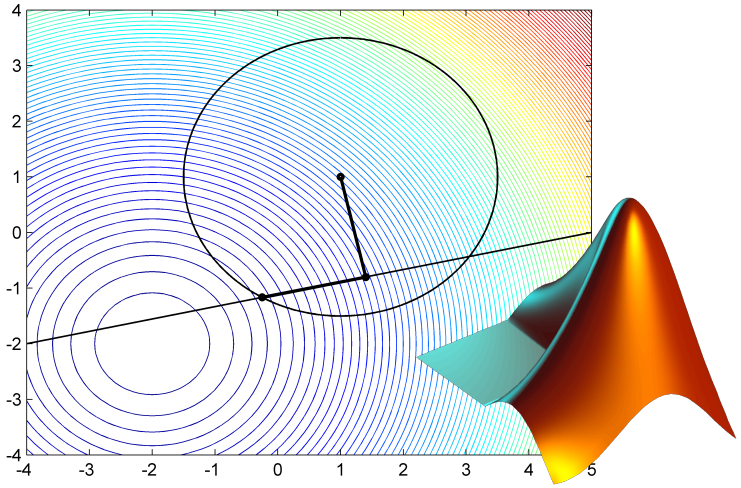
Wie entwirft man Raumschiffe?

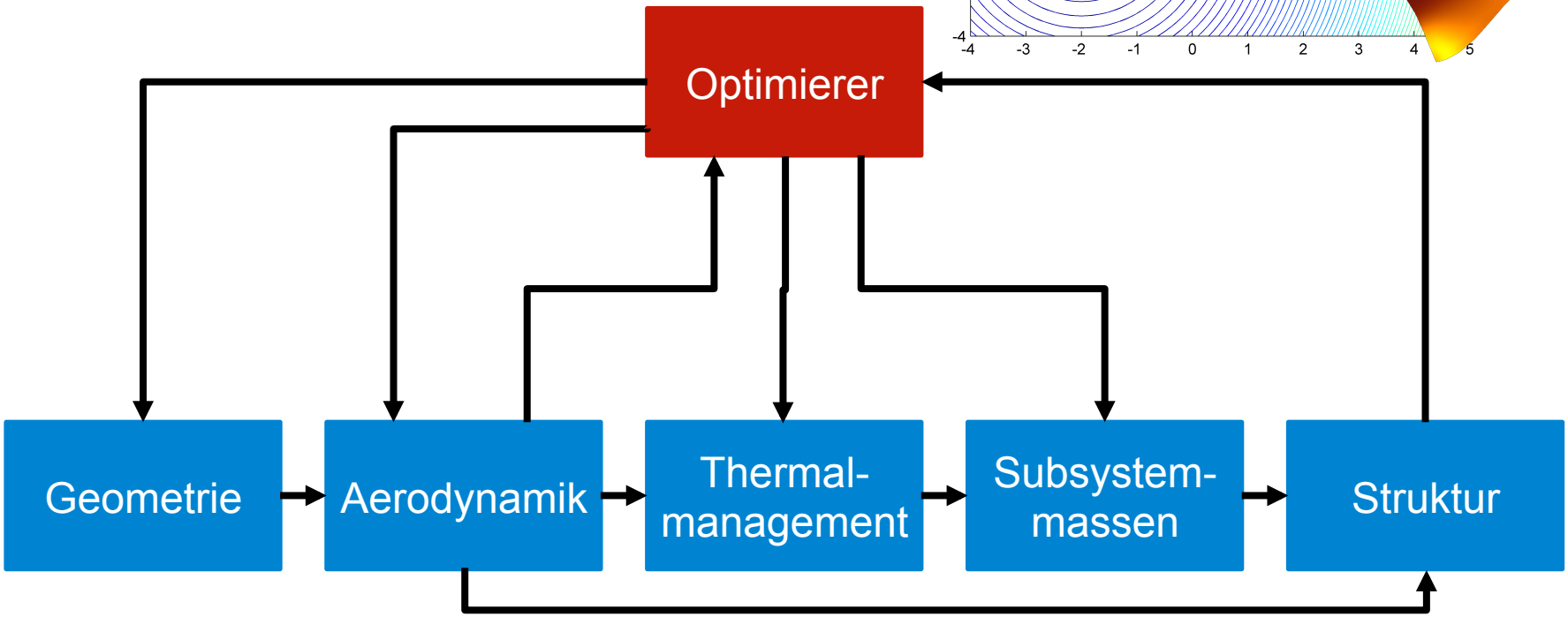


Simulations-Workflow
Vernetzung der Fachdisziplinen



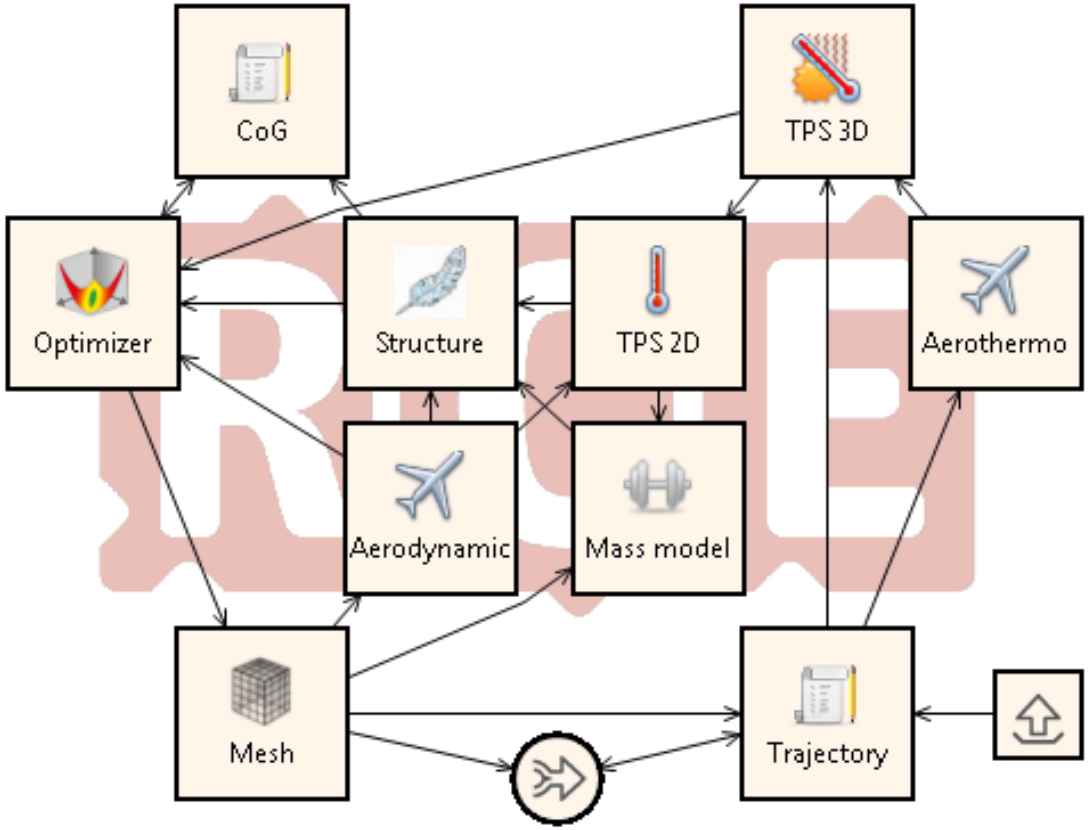
Simulations-Workflow
Mit Optimierung



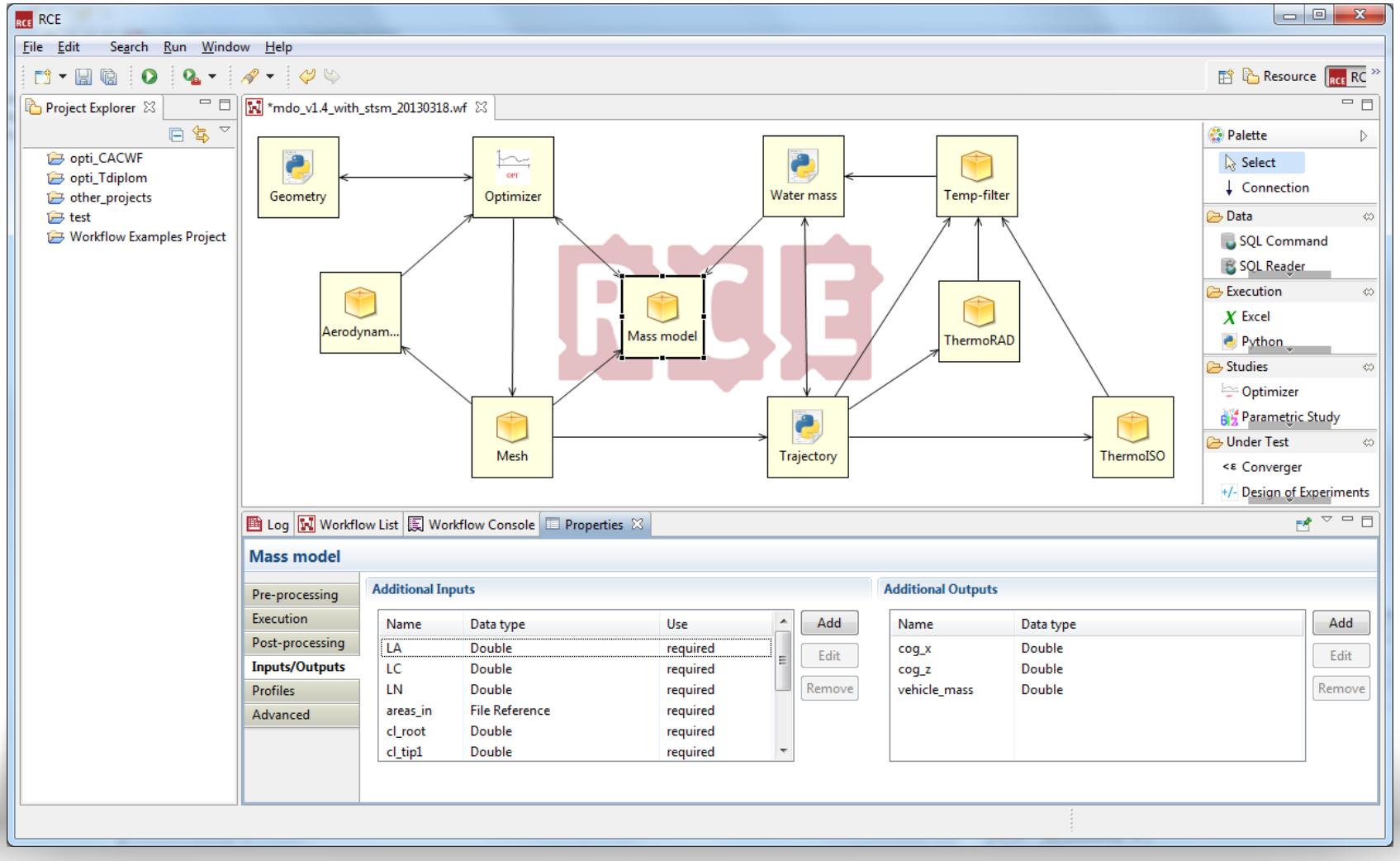


Simulations-Workflow

Integration in eine Simulationsumgebung



Simulationsumgebung RCE



Vielen Dank!

Fragen?

Andreas.Schreiber@dlr.de

www.DLR.de/sc | @onyame